OFFICE OF THE SECRETARY OF DEFENSE
OPERATIONAL TEST AND EVALUATION
1700 DEFENSE PENTAGON
WASHINGTON, DC 20301-1700

DEPARTMENT OF THE ARMY
US ARMY MATERIEL COMMAND
ABERDEEN PROVING GROUND, MD 21005-5071

DEPARTMENT OF THE NAVY
ASSESSMENT DIVISION OPNAV (N81)
WASHINGTON, DC 20350-2000

DEPARTMENT OF THE AIR FORCE
HEADQUARTERS AIR ARMAMENT CENTER (AFMC)
EGLIN AFB, FL 32542-6808

JOINT TECHNICAL COORDINATING GROUP FOR MUNITIONS EFFECTIVENESS
ABERDEEN PROVING GROUND, MARYLAND 21005-5071

RDAM-J 5 Aug 10

MEMORANDUM FOR Joint Technical Coordinating Group for Munitions Effectiveness Product Management Office (OC-ALC/ENLB/Ms. Sandra Hysell), 7851 Arnold Street, Suite 202, Tinker AFB, OK 73145-9160

SUBJECT: Distribution Statement for 61 JTCG/ME-71-7-1, 61 JTCG/ME-7-2-1 and 61 JTCG/ME-71-7-2-2

1. A review of the subject Magic Computer Simulation User and Analyst Manuals has been conducted based upon a request received from the US Army Research Laboratory. This review resulted in the decision to release these publications into the public domain. As such, request the following distribution statement be added to these items: "Approved for public release; distribution is unlimited."

2. Request, therefore, recipients of these publications be notified of this distribution statement.

3. The JTCG/ME Program Office point of contact for this request is Mrs. Chantal B. Marus, COMM (410) 278-6740, DSN 298-6740; e-mail: chantal.b.marus@us.army.mil.

BRYAN W. PARIS
Director
JTCG/ME Program Office

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **MAY 1971**	2. REPORT TYPE **Final**	3. DATES COVERED **00-00-1969 to 00-00-1971**
4. TITLE AND SUBTITLE **MAGIC Computer Simulation Analyst Manual Part 2**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Weapons Center,China Lake,CA,93555**		8. PERFORMING ORGANIZATION REPORT NUMBER **TN4565-3-71 Vol II**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) **Joint Technical Coordinating Group for Munitions Effectiveness, ABERDEEN PROVING GROUND, MD, 21005-5071**		10. SPONSOR/MONITOR’S ACRONYM(S)
		11. SPONSOR/MONITOR’S REPORT NUMBER(S) **61 JTCG/ME-71-7-2-2**

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The MAGIC computer simulation generates target descripfion data consisting of item-by-item listings of the target’s components and air- spaces encountered by a large number of parallel rays emanating from any desired attack angle. A combinatorial geometry technique, which defines the locations and shapes of the various physical regions in terms of the intersections and unions of the volumes contained in a set of simple bodies, is used to represent complex target structures. A grid ce1l pattern is superimposed over the surface of the target and parallel rays are "fired" from each grid cell.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**268**	

61 JTCG/ME-71-7-2-2

JTCG/ME

MAGIC COMPUTER SIMULATION

VOLUME II. ANALYST MANUAL

PART II

Produced for:

Joint Technical

Coordinating Group

for

Munitions Effectiveness

MAY 1971

ABSTRACT

The MAGIC computer simulation generates target description data consisting of item-by-item listings of the target's components and air-spaces encountered by a large number of parallel rays emanating from any desired attack angle. A combinatorial geometry technique, which defines the locations and shapes of the various physical regions in terms of the intersections and unions of the volumes contained in a set of simple bodies, is used to represent complex target structures. A grid cell pattern is superimposed over the surface of the target and parallel rays are "fired" from each grid cell.

Volume II, Part II contains:

Section III	Simulation Model, Subroutine QRTIC through Subroutine TESTG; and List of Symbols and Abbreviations.
Section IV	Source Listing

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

Figure | Page

Part I (Concluded)

Part II

LIST OF FIGURES (Concluded)

Subroutine QRTIC(C,R,N)

Subroutine QRTIC is called by Subroutine TOR to solve a polynomial equation of the type $x^4 + ax^3 + bx^2 + cx + d = 0$, using Ferrari's solution of the quartic equation where the coefficient of x^4 is assumed to be one. Four-element array R will contain the roots, and variable N will contain the number of real roots. If there are two real roots, they will be placed in R(1) and R(2) with the complex roots in R(3) and R(4) in the form R(3) ± iR(4). If there are no real roots, the complex roots will be stored in the form R(1) ± R(2) and R(2) ± iR(4). Subroutine QRTIC calls Subroutine CUBIC to find the roots of the resolvent cubic equation.

The statement

```
      DIMENSION C(4),R(4),CC(3),RR(3)
```

is used to dimension arrays for the four coefficients of the quartic equation, for the four roots of the quartic equation, for the three coefficients of the resolvent cubic equation, and for three roots of the cubic equation.

The statements

```
C
C1      SOLVES A POLYNOMIAL EQUATION OF THE TYPE X**4 + C(1)*X**3
C       + C(2)*X**2 + C(3)*X + C(4) = 0 USING THE FERRARI SOLUTION OF
C       THE QUARTIC EQUATION. THE COEFFICIENT OF X**4 IS ASSUMED TO BE 1.
C       R(4) CONTAINS THE ROOTS. N CONTAINS THE NUMBER OF REAL ROOTS.
C       IF THERE ARE 2 REAL ROOTS THEY WILL BE IN R(1) AND R(2), WITH THE
C       COMPLEX ROOTS IN R(3)+- R(4)*I. IF THERE ARE NO REAL ROOTS THE
C       COMPLEX ROOTS ARE IN R(1)+-R(2)*I AND R(3)+-R(4)*I.
C
C       COMPUTE RESOLVENT CUBIC
C
        C1SQ=C(1)*C(1)
        CC(1)=-.5*C(2)
        CC(2)=.25*C(1)*C(3)-C(4)
        CC(3)=.125*( C(4)*(4.*C(2)-C1SQ)-C(3)*C(3) )
        CALL CUBIC(CC,RR,NN)
```

are used to compute the coefficients of the resolvent cubic equation

$$\boxed{w^3 - \frac{b}{2} w^2 + \frac{(ac-4d)}{4} w + \frac{4bd-a^2d-c^2}{8} = 0} \qquad (201)$$

where y = 2w, from the coefficients of the quartic equation. Subroutine CUBIC is called to compute the roots of the resolvent cubic equation.

The statement

```
C
C>      DETERMINE IF POSSIBLE SOLUTION
C
        T=.25*C1SQ-C(2)
```

is used to compute the quantity $a^2/4-b$ for use in solving the coefficients of

$$\boxed{\left(\frac{a^2}{4}-b+2w\right)x^2 + (aw-c)\;x + (w^2-d) = e^2\;x^2 + 2efx + f^2} \tag{202}$$

which splits into two quadratic equations with properly chosen e and f:

$$x^2 + \frac{a}{2}x + \frac{1}{2}y = \pm(ex+f)$$

Solving for x using the quadratic formula results in

$$\boxed{x_{1,2} = \frac{-\frac{a}{2}+e}{2} \pm \frac{\sqrt{\left(-\frac{a}{2}+e\right)^2 - 4\;(w-f)}}{2}} \tag{203}$$

for two roots and

$$\boxed{x_{3,4} = \frac{-\frac{a}{2}-e}{2} \pm \frac{\sqrt{\left(-\frac{a}{2}-e\right)^2 - 4\;(w-f)}}{2}} \tag{204}$$

for the remaining two roots.

The statement

```
        DO 10 I=1,NN
```

is used to begin a DO loop to test each real root returned from Subroutine CUBIC to determine if there are any real roots in the quartic equation.

The statements

```
        ROOT=RR(I)
        ASQ=T+ROOT*ROOT
        IF(ABS(ASQ).LE.0.000001)ASQ=0.
        IF(ASQ.LT.0.0)GOTO 10
```

are used to compute the coefficient of x^2 of Equation (202). If it is very nearly zero, it is set to zero. The coefficient is tested for a less-than-zero condition, which means that for the given real root of the cubic equation there are no real roots for the quartic equation. Therefore, the program loops to consider the next real root, if any.

The statements

```
      BSQ=ROOT*ROOT-C(4)
      IF(ABS(BSQ).LE.0.000001)BSQ=0.
      IF(BSQ.GE.0.0)GOTO 20
   10 CONTINUE
      N=0
      RETURN
```

are used to compute the constant term of Equation (202). If it is very nearly zero it is set to zero. The value is tested for a greater than or equal to zero condition, which means that for the given real root of the cubic equation there are real roots for the quartic equation. If the constant term of Equation (202) is less than zero, the given real root of the cubic equation will not result in real roots for the quartic equation. The program therefore loops to consider the next real cubic root. If all real cubic roots have been considered without satisfying the conditions for real roots in the quartic equation, the variable N (the number of real roots in the quartic equation) is set to zero, and the subroutine returns control to Subroutine TOR.

The statements

```
C
C     COMPUTE FIRST TWO ROOTS OF QUARTIC EQUATION
C
   20 TWOAB=C(1)*ROOT-C(3)
      A=SQRT(ASQ)
      B=SIGN(SQRT(BSQ),TWOAB)
      N=0
```

are used to compute the coefficient of x, the square root of the coefficient of x^2, and the square root of the constant term with the sign of the coefficient of x of Equation (202). The variable N (the number of real roots in the quartic equation) is initialized to zero.

The statements

```
REAL=.25*(A*A-C(1))
DISC=REAL*REAL-ROOT+B
SQROOT=SQRT(ABS(DISC))
```

are used to compute the values for solving quadratic Equation (203) for the first set of roots; and to solve for the value of the square root of the discriminate.

The statements

```
IF(ABS(DISC).LE.0.000001)DISC=0.
IF(DISC.LT.0.0)GOTO 30
```

are used to determine if the absolute value of the discriminate is very nearly equal to zero, and, if it is, to set it to zero. A test is made to determine if the value of the discriminate is less than zero. If it is, the roots are complex conjugates. Therefore, the program branches to compute the two imaginary roots.

The statements

```
C
C4     DISCRIMINATE .GE. 0  COMPUTE 2 REAL ROOTS
C
       N=2
       R(1)=REAL+SQROOT
       R(2)=REAL-SQROOT
       GOTO 40
```

are executed if the discriminate was greater than or equal to zero. They are used to set variable N to two to indicate two real roots. The values of the roots are computed and stored in the first two elements of array R. The program then branches to determine the condition of the two remaining roots.

The statements

```
C
C5     DISCRIMINATE .LT. 0  COMPUTE 2 IMAGINARY ROOTS
C
    30 R(3)=REAL
       R(4)=SQROOT
```

are executed if the discriminate was less than zero, indicating that the roots are complex conjugates. These statements therefore store the imaginary roots in the last two elements of array R.

The statements

```
C
C6     COMPUTE LAST TWO ROOTS OF QUARTIC EQUATION
C
   40 REAL=REAL-A
      DISC=REAL*REAL-ROOT-B
      SQROOT=SQRT(ABS(DISC))
```

are used to compute the values for solving quadratic Equation (204) for the second set of roots and to solve for the value of the square root of the discriminate.

The statements

```
      IF(ABS(DISC).LE.0.000001)DISC=0.
      IF(DISC.LT.0.0)GOTO 50
```

are used to determine if the absolute value of the discriminate is very nearly equal to zero, and, if it is, to set it to zero. A test is made to determine if the value of the discriminate is less than zero. If it is, the roots are complex conjugates, and the program branches to store the two imaginary roots.

The statements

```
C
C7     DISCRIMINATE .GE. 0  COMPUTE 2 REAL ROOTS
C
       N=N+2
       R(N)=REAL-SQROOT
       R(N-1)=REAL+SQROOT
       RETURN
```

are executed if the previous test of the discriminate proved that the roots are real. These statements store the two real roots in the R array after determining the proper location in the R array based on the number of real roots already computed. Control is then returned to Subroutine TOR with the values of the real roots.

The statements

```
C
CA     DISCRIMINATE .LT. 0  COMPUTE 2 IMAGINARY ROOTS
C
   50 R(N+1)=REAL
      R(N+2)=SQROOT
      RETURN
```

are executed if the discriminate was less than zero, indicating that the roots are complex conjugates. These statements therefore store the imaginary roots in the R array at a location based on the number of real roots already computed. Control is then returned to Subroutine TOR with the values of the real roots.

Subroutine CUBIC(C,R,N)

Subroutine CUBIC is called by Subroutine QRTIC to solve a polynomial equation of the type $x^3 + ax^2 + bx + c = 0$ where the coefficient of x^3 is assumed to be one. The three roots are passed back to Subroutine QRTIC through argument R. The number of real roots is passed back through argument N. If there is only one real root, it will be in R(1) with the complex roots in R(2) i*R(3).

The statement

```
      DIMENSION C(3),R(3)
```

is used to dimension two three-element arrays for the three coefficients of the polynomial equation and the three roots of the equation.

The statements

```
C
C1    COMPUTE ROOTS OF CUBIC EQUATION
C
      C1SQ=C(1)*C(1)
      P=C(2)-C1SQ/3.
      Q=C(3)+C(1)*(2.*C1SQ/27.-C(2)/3.)
```

are used to solve the expressions (from Cardan's Solution)

$$\boxed{p = b - \frac{a^2}{3}} \tag{207}$$

and

$$\boxed{q = c + a \left(2 \frac{a^2}{27} - \frac{b}{3}\right)} \tag{208}$$

where c(1) = a, c(2) = b, and c(3) = c

The statements

```
      DISC=4.*P*P*P+27.*Q*Q
      C3=C(1)/3.
```

are used to solve the expression

$$\boxed{Q = \left(\frac{p}{3}\right)^3 + \left(\frac{q}{2}\right)^2} \tag{214}$$

from the equation

$$\boxed{Q = \left(\frac{p}{3}\right)^3 + \left(\frac{q}{2}\right)^2} \qquad (214)$$

and a/3 from the equation

$$\boxed{x = y - \frac{a}{3}} \qquad (209)$$

The statements

```
IF(ABS(DISC).LE.1.0E-8)DISC=0.
IF(DISC.LE.0.0)GOTO 10
```

are used to set DISC to zero if the absolute value of DISC is very nearly zero. DISC is tested to determine if it is less than or equal to zero. If it is, the program branches to compute three real roots for the polynomial equation. If DISC is greater than zero, there are only one real root and two complex roots.

The statement

```
C
C2    CONDITION FOR 1 REAL AND 2 COMPLEX ROOTS
C
      N=1
```

is used to set N to one to indicate to the calling program that the solution of the cubic equation results in only one real root.

The statements

```
SQROOT=SQRT(DISC/108.)
HALFQ=.5*Q
ACU=-HALFQ+SQROOT
BCU=-HALFQ-SQROOT
```

are used to solve the quantities under the radical of equations

$$\boxed{A = \sqrt[3]{-\frac{q}{2} + \sqrt{Q}}} \qquad (212)$$

and $$\boxed{B = \sqrt[3]{\frac{-q}{2} - \sqrt{Q}}} \qquad (213)$$

The statements

```
A=SIGN(ABS(ACU)**.3333333333333333,ACU)
B=SIGN(ABS(BCU)**.3333333333333333,BCU)
```

are used to solve Equations (212) and (213) for A and B, respectively, using the sign of the expression under the radical as the sign of the result.

The statement

```
AB=A+B
```

is used to compute the value of A + B for solving the equations

$$\boxed{y_1 = A + B} \qquad (210)$$

$$\boxed{y_{2,3} = -\frac{A+B}{2} \pm i\ \frac{A-B}{2}\sqrt{3}} \qquad (211)$$

The statements

```
R(1)=AB-C3
R(2)=-.5*AB-C3
R(3)=.866025404*(A-B)
RETURN
```

are used to compute the one real root, with the result in R(1), and the two conjugate complex roots, with the real part of the complex root in R(2) and the complex part in R(3) using Equations (212) and (213) and Equation (211). Program control returns to Subroutine QRTIC.

The statement

```
C
C3    CONDITION FOR 3 REAL ROOTS
C
   10 N=3
```

is used to set N to three to indicate to the calling program that the Solution of the cubic equation results in three real roots.

The statements

```
T=SQRT(ABS(P)/3.)
TT=T+T
```

are used to compute the value of $2\sqrt{-p/3}$ for use in solving the cubic equations (trigonometric solution)

$$\boxed{y_1 = 2\sqrt{-p/3}\cos(\phi/3)} \tag{215}$$

$$\boxed{y_{2,3} = -2\sqrt{-p/3}\cos(\phi/3 \pm 60^o)} \tag{216}$$

The statement

```
IF(DISC.EQ.0.0)GOTO 20
```

is used to determine if two of the three real roots are equal by testing DISC for a zero value.

The statement

```
PHI3=ATAN2(SQRT(-DISC/27.),-Q)/3.
```

is used to compute the value of $\Phi/3$ in radians for solving Equations (215) and (216) where

$$\boxed{\cos\phi = \frac{q}{2\sqrt{-(p/3)^3}}} \tag{217}$$

The statements

```
R(1)=TT*COS(PHI3)-C3
R(2)=TT*COS(PHI3+2.094395103)-C3
R(3)=TT*COS(PHI3-2.094395103)-C3
RETURN
```

are used to compute the three unequal roots from Equations (215) and (216) and Equation (217). The program returns control to Subroutine QRTIC.

The statements

```
C
C4     CONDITION FOR 2 OR 3 EQUAL ROOTS
C
   20 R(1)=SIGN(TT,-Q)-C3
      R(2)=SIGN(T,Q)-C3
      R(3)=R(2)
      RETURN
```

are executed if two of the three real roots are equal. One root is equal to $2\sqrt{-(p/3)} = y$ in Equation (215) with the sign of y equal to the sign of -q. The other two equal roots are equal to $\sqrt{-(p/3)} = y$ in Equation (216) with the sign of y equal to the sign of q. The program returns control to Subroutine QRTIC.

Subroutine UN2(L,J1,J2)

Subroutine UN2 is called by other subroutines of the MAGIC program when unpacking of two integer data items from a single computer word is required. L, the pointer to the location of the word in the MASTER array, is passed to Subroutine UN2. Subroutine UN2 unpacks the two 15-bit integer data items and passes them back to the calling routine via Subroutine UN2 integer variables J1 and J2. The two data items appear in packed format as:

	15 bits	15 bits
Unused	J1	J2

↑
L

The statement

```
C
C      UNPACK 2 15-BIT INTEGER DATA ITEMS FROM L WORD IN MASTER ARRAY
C
       COMMON MASTER(10000)
```

is used to make the MASTER array available to this subroutine.

The statements

```
       I3=MASTER(L)
       J1=I3/32768
       J2=I3-J1*32768
       RETURN
```

are used to store the MASTER array word referenced by pointer L in word I3. Variable J1 is equated to the data in the 15 bits previous to the last 15 bits by dividing the word by 2^{15}, thus shifting right 15 bits and leaving only J1. The J2 data item is unpacked by subtracting J1 left-shifted by 15 bits from I3, leaving only J2. The left-shift of 15 bits is performed by multiplying the word by 2^{15}.

Subroutine UN3(L,J1,J2,J3)

Subroutine UN3 is called by Subroutine G1 when unpacking of three integer data items from LIO, Subroutine G1 working storage in the MASTER array is required. L, the location of the packed word, is passed to Subroutine UN3 from Subroutine G1. Subroutine UN3 unpacks the two six-bit and one 15-bit integer data items and passes them back to Subroutine G1 via Subroutine UN3 integer variables J1, J2, and J3. The three data items appear in packed format as:

	6 bits	6 bits	15 bits
Unused	J1	J2	J3

↑
L

The statement

```
C
C      UNPACK 2 6-BIT AND 1 15-BIT INTEGER DATA ITEMS FROM G1 WORKING
C      STORAGE AT THE L WORD IN THE MASTER ARRAY
C
       COMMON MASTER(10000)
```

is used to make the MASTER array available to this subroutine.

The statements

```
       I3=MASTER(L)
       I2=I3/32768
       J1=I2/64
       J2=I2-J1*64
       J3=I3-I2*32768
       RETURN
```

are used to store the MASTER array word referenced by pointer L in word I3. I3 is right-shifted 15 bits and equated to I2, leaving only J1 and J2. J1 is determined by right-shifting I2 six bits, leaving only the six bits of J1. J2 is determined by subtracting J1, left-shifted by six bits, from I2, leaving only J2. J3 is finally determined by subtracting J1 and J2 of I2, left-shifted by 15 bits, from I3, leaving only J3. The right-shift of 6 bits or 15 bits is performed by dividing word L by 2^6 or 2^{15}, respectively. The left-shift of 6 bits or 15 bits is performed by multiplying word L by 2^6 or 2^{15} respectively.

Subroutine OPENK(L,J1,J2,J3)

Subroutine OPENK is called by Subroutine CALC when unpacking of three integer data items from the ITR array prepared by Subroutine TRACK is required. L, pointer to the location of the word in the ITR array, is passed to this subroutine from Subroutine CALC. Subroutine OPENK unpacks the three 12-bit integer data items and passes them back to Subroutine CALC via integer variables J1, J2, and J3. The three data items appear in packed format as:

	12 bits	12 bits	12 bits
Unused	J1	J2	J3

↑
L

The statement

```
      COMMON/GTRACK/D1,D2,KHIT,LMAX,TR(200),XBS(3),IRSTRT,IENC,
     1    ITR(200),CA,CE,SA,SE
```

is used to make the ITR array available to this subroutine.

The statements

```
C
C1    UNPACK 3 12-BIT INTEGER DATA ITEMS FROM COMPONENT LINE-OF-SIGHT
C     STORAGE ARRAY ITR. THE THREE ITEMS ARE
C        / SURFACE NUMBER /  BODY NUMBER   /   NEXT REGION  /
C
      I3=ITR(L)
      I2=I3/4096
      J1=I2/4096
      J2=I2-J1*4096
      J3=I3-I2*4096
      RETURN
```

are used to store the ITR array word referenced by L in word I3. I3 is right-shifted 12 bits and equated to I2, leaving only J1 and J2 in I2. J1 is determined by right-shifting I2 12 bits, leaving only the 12 bits of J1. J2 is determined by subtracting J1, left-shifted by 12 bits, from I2, leaving only J2. J3 is finally determined by subtracting J1 and J2 of I2, left-shifted by 12 bits, from I3, leaving only J3. The left-shift of word L by 12 bits is performed by multiplying by 2^{12}. The right-shift is performed by dividing word L by 2^{12}.

Function RAN(M)

Function RAN is called by Subroutines GRID, AREA, and TROPIC to generate a random number between zero and one.

The statement

```
COMMON/RANDM/IRN
```

is used to pass a number into this routine from which a number between zero and 0.9 will be generated. A new number is passed back into the common area for use when the random function is again called.

The statements

```
C
C1    GIVEN A NUMBER IRN. GENERATE A RANDOM NUMBER BETWEEN 0 AND 1
C
      RAN=URAN31(IRN)
      RETURN
```

are used to call Function URAN31 to generate a random number from the number given in IRN and to return control to the calling subroutine when the random number has been determined.

Function URAN31(I)

Function URAN31 is called by Function RAN to perform the actual computations of the generating of a random number between zero and 0.9. The argument is also revised.

The statements

```
   IF(I)20,10,20
10 I=11111111
```

are used to determine if the argument for Functions RAN and URAN31 has a value other than zero. If the argument is zero, the number 11111111 is assigned to the argument.

The statements

```
20 J=I
   J=J*25
   J=J-(J/67108864)*67108864
   J=J*25
   J=J-(J/67108864)*67108864
   J=J*5
   J=J-(J/67108864)*67108864
   A1=J
   I=J
   URAN31=A1/67108864.
   RETURN
```

are used to generate a random number from the number 11111111 if the function argument is zero; or from the value of the argument if it is other than zero. The resultant random number between zero and 0.9 is assigned to the function name; at the same time a new value for the argument is assigned for future calls to the random function.

Subroutine CROSS(ANSWER,FIRST,SECOND)

Subroutine CROSS can be called by Subroutine GENI, CALC, or ARS for the purpose of computing the coordinates of the resultant vector from the cross product of two vectors.

The statement

```
DIMENSION ANSWER(3),FIRST(3),SECOND(3)
```

is used to dimension three three-element arrays for the coordinates of the two known vectors and the resultant cross-product vector.

The statements

```
C
C      COMPUTE CROSS PRODUCT     ANSWER = FIRST X SECOND
C
       ANSWER(1) = FIRST(2)*SECOND(3) - FIRST(3)*SECOND(2)
       ANSWER(2) = FIRST(3)*SECOND(1) - FIRST(1)*SECOND(3)
       ANSWER(3) = FIRST(1)*SECOND(2) - FIRST(2)*SECOND(1)
       RETURN
```

are used to compute the cross-product of two vectors whose coordinates are passed by arguments FIRST and SECOND; and to store the coordinates of the resultant vector in argument ANSWER for passing back to the calling program. The cross product of two vectors $\bar{A}$ and $\bar{B}$ is given by the expression

$$\bar{A} \times \bar{B} = (\bar{A}y \cdot \bar{B}z - \bar{A}z \cdot \bar{B}y)_x + (\bar{A}z \cdot \bar{B}x - \bar{A}x \cdot \bar{B}z)_y + (\bar{A}x \cdot \bar{B}y - \bar{A}y \cdot \bar{B}x)_z$$

Function DOT(FIRST,SECOND)

Function DOT is called by Subroutines GENI, CALC, ARS, TEC, and TOR for computing the resultant scalar quantity from the dot product of vectors FIRST and SECOND.

The statement

```
      DIMENSION FIRST(3),SECOND(3)
```

is used to dimension two three-element arrays for the coordinates of the two vectors passed in the arguments.

The statements

```
C
C1    COMPUTE DOT PRODUCT     DOT = FIRST . SECOND
C
      DOT = FIRST(1)*SECOND(1)+FIRST(2)*SECOND(2)+FIRST(3)*SECOND(3)
      RETURN
```

are used to compute the dot product of two vectors whose coordinates are passed by function arguments FIRST and SECOND. The resultant scalar quantity is returned to the calling program via the function name. The dot product of two vectors $\overline{A}$ and $\overline{B}$ is given by the expression

$$\overline{A}\cdot\overline{B} = \overline{A}x\cdot\overline{B}x + \overline{A}y\cdot\overline{B}y + \overline{A}z\ \overline{B}z$$

Subroutine UNIT(V)

Subroutine UNIT is called by Subroutine GENI or Subroutine CALC to compute the direction cosines of a vector passed to this subroutine by argument V. The resultant coordinates are passed back to the calling program through argument V.

The statement

```
      DIMENSION V(3)
```

is used to dimension a three-element array for the coordinates of the vector passed to this subroutine.

The statements

```
C
C1    COMPUTE UNIT VECTOR (DIRECTION COSINES OF VECTOR)
C
      TEMP = SQRT(DOT(V,V))
      V(1)=V(1)/TEMP
      V(2)=V(2)/TEMP
      V(3)=V(3)/TEMP
      RETURN
```

are used to compute the scalar length of the vector by computing the square root of the vector dotted on itself. The length of each coordinate vector is divided by the scalar length of the vector, resulting in the direction cosines of the vector. These are returned to the calling program through argument V. The unit vector of a given vector $\bar{A}$ can be expressed mathematically as

$$\bar{A}_i = \frac{\bar{A}_i}{\sqrt{\bar{A} \cdot \bar{A}}}$$

where i is the x, y, or z coordinate.

Function XDIST(XA,XB)

Function XDIST is called by Subroutines TRACK, CALC, TESTG, VOLUM, and DCOSP to compute the distance between two points passed into this routine through arguments $\overline{XA}$ and $\overline{XB}$. This function subprogram computes the distance by utilizing the standard distance formula

$$\text{Distance} = \sqrt{\sum(\overline{XA} - \overline{XB})^2}$$

The statements

```
C
C1    COMPUTE THE DISTANCE BETWEEN TWO GIVEN POINTS XA AND XB
C
      DIMENSION XA(3),XB(3)
      XSUM=0.
```

are used to dimension two three-element arrays for the coordinates of the two points passed into this routine. Variable XSUM is then initialized to zero.

The statements

```
      DO 10 I=1,3
      XSUM=XSUM+(XA(I)-XB(I))**2
   10 CONTINUE
```

consist of a DO loop which is used to sum the distance squared for each of the coordinates between the two points.

The statements

```
      XDIST=SQRT(XSUM)
      RETURN
```

are used to compute the square root of the summed distance squared which results in the distance between the two given points. The routine returns to the calling program.

Subroutine DCOSP(XA,XB,WA)

Subroutine DCOSP is called by Subroutines CALC, TESTG, and VOLUM to compute the direction cosines from point $\overline{XA}$ to point $\overline{XB}$. This is accomplished by calling Function XDIST to compute the scalar distance from point $\overline{XA}$ to $\overline{XB}$; and dividing each coordinate vector between the two points by the distance between the two points.

The statement

```
C
C1    COMPUTE THE DIRECTION COSINES FROM POINT XA TO POINT XB
C     AND STORE DIRECTION COSINES IN WA
C
      DIMENSION XA(3),XB(3),WA(3)
```

is used to dimension three three-element arrays for the coordinates of the two points and the resultant direction cosines.

The statement

```
      DIS=XDIST(XA,XB)
```

is used to call Function XDIST to compute the distance between the two given points passed to this subroutine.

The statements

```
      DO 10 I=1,3
      WA(I)=(XB(I)-XA(I))/DIS
   10 CONTINUE
      RETURN
```

consist of a DO loop which is used to divide each coordinate vector between the two points by the distance between the two points. The result is the direction cosines of the vector from point $\overline{XA}$ to point $\overline{XB}$. In mathematical terms

$$\overline{WA}_i = \frac{\overline{XB}_i - \overline{XA}_i}{\sqrt{\sum(XA_i - \overline{XB}_i)^2}}$$

where i represents the x, y, or z coordinate of the point.

Subroutine TROPIC(WP)

Subroutine TROPIC is called by Subroutines GRID and AREA to calculate isotropic direction cosines. Each call to Subroutine TROPIC generates a different set of direction cosines.

The statement

```
C
C1    GENERATE RANDOM DIRECTION COSINES FROM AN ISOTROPIC DISTRIBUTION
C
      DIMENSION WP(3)
```

is used to dimension a three-element array for the coordinates of the generated random direction cosines.

The statements

```
   10 X1=RAN (-1)
      X2=RAN (-1)
      X1S=X1**2
      X2S=X2**2
      T=X1S+X2S
      IF(T.GE.1.)GOTO 10
```

are used to call function RAN to generate random numbers X1 and X2 and to compute the sum of the squares. A test is made to determine if $X1^2 + X2^2 \leq 1.0$. If not, the statements are repeated until the expression is satisfied.

The statements

```
C
C2    COMPUTE THE SINE AND COSINE OF A RANDOM ANGLE PHI
C
      CSPHI=(X1S-X2S)/T
      SNPHI=(2.*X1*X2)/T
```

are used to compute the cosine of random angle ϕ from the expression

$$\cos \phi = \frac{X1^2 - X2^2}{X1^2 + X2^2}$$

and to compute the sine of random angle ϕ from the expression

$$\sin \phi = \frac{2.0 \cdot X1 \cdot X2}{X1^2 + X2^2}$$

The statements

```
      X1=RAN (-1)
      IF(X1.LE..5)SNPHI=-SNPHI
```

are used to call Function RAN to compute a random number between zero and one and to set sin ϕ negative if the random number was less than or equal to 0.5.

The statements

```
C
C     COMPUTE THE SINE AND COSINE OF A RANDOM ANGLE THETA
C
      CSTHT=2.*RAN (-1)-1.
      SNTHT=SQRT(1.-CSTHT**2)
```

are used to call Function RAN to compute a random number between zero and one, double it, and subtract one to arrive at the cosine of random angle θ. The sine of θ is computed from the expression

$$\sin \theta = \sqrt{1 - (\cos \theta)^2}$$

The statements

```
C
C     COMPUTE RANDOM DIRECTION COSINES
C
      WP(1)=SNTHT*SNPHI
      WP(2)=SNTHT*CSPHI
      WP(3)=CSTHT
      RETURN
```

are used to compute the x, y, and z coordinates of the random direction cosines and store them in three-element array WP via the following expressions:

$$\overline{WP}(1) = \sin \theta \cdot \sin \phi$$
$$\overline{WP}(2) = \sin \theta \cdot \cos \phi$$
$$\overline{WP}(3) = \cos \theta$$

the program returns to the calling program with WP.

Function S(I,N)

Function S is called by Subroutines RPPIN, RPP2, and RPP for the purpose of retrieving the coordinate of any one of the six sides of a rectangular parallelepiped (RPP). Given argument I, the ordinal number of the RPP, and argument N, the side number of the RPP where N = 1, 2, 3, 4, 5, or 6 (Xmin, Xmax, Ymin, Ymax, Zmin, or Zmax respectively), the routine will compute the location in the ASTER array of the required coordinate and return with the value of the coordinate set equal to S.

The statements

```
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      EQUIVALENCE(MASTER,ASTER)
```

are used to dimension the MASTER array for 30,000 words, to make the data in the ASTER array available to this routine, to pass the value of pointer LBASE into this routine, and to set the MASTER array equivalent to the ASTER array.

The statements

```
C
C1    S RETRIEVES COORDINATE OF ANY ONE OF THE 6 SIDES OF AN RPP
C         I IS THE RPP NUMBER      N IS THE SURFACE NUMBER
C
      L=LBASE+12*(I-1)+2*(N-1)
      LL=MASTER(L+1)
      S=ASTER(LL)
      RETURN
```

are used to compute the location of the pointer data for the given side of the given RPP and to retrieve the pointer from the next word since that is the location of the pointer to the coordinate for the given side. Using this pointer, the coordinate is retrieved from the ASTER array and equated to S. The routine returns to the calling program.

Subroutine RPP2(LSURF,XP,IRP)

Subroutine RPP2 is called by Subroutine G1 whenever an exit intersect occurs with a surface of the RPP containing the target geometry. The purpose of the subroutine is to determine the number of the RPP (if any) that the ray is entering upon leaving the present RPP. If no abutting RPP can be found, the routine returns a zero as the number of the abutting RPP.

The statement

```
C
C1    FIND NUMBER OF ABUTTING RPP TO INTERSECTED SURFACE
C
      DIMENSION XP(3)
```

is used to dimension a three-element array to contain the coordinates of the exit intersect point of the present RPP.

The statements

```
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
```

are used to make the contents of the ASTER array available to this routine and to pass the RPP number and the beginning location of the RPP data into this routine.

The statements

```
C
      LOC=LBASE+12*(NASC-1)-2*(LSURF+1)
      CALL UN2(LOC,LOCAT,NC)
```

are used to compute the location of the pointer data for the surface number of the RPP that the ray is exiting; and to unpack the number of abutting RPP's to that surface and the pointer to the location in the MASTER array of the list of abutting RPP's.

The statements

```
      IF(NC-1)10,20,30
```

```
10 IRP=0
   RETURN
```

are used to determine if there are no abutting RPP's, only one abutting RPP, or more than one abutting RPP to the intersected surface of the RPP. If there are no abutting RPP's, argument IRP is set to zero, and control is returned to Subroutine G1.

The statements

```
20 CALL UN2(LOCAT,IRP,DUM)
   RETURN
```

are executed if there is only one abutting RPP to the intersected surface. These statements therefore unpack the abutting RPP from its location in the abutting RPP section and return control to Subroutine G1 with argument IRP set to the number of the abutting RPP.

The statement

```
30 M=1
```

is executed if there is more than one abutting RPP to the intersected surface. Variable M is set to one and is subsequently used to determine which of two abutting RPP's in a packed word is to be tested.

The statements

```
C
      DO 90 I=1,NC
      M=-M
```

are used to begin a DO loop which will test each abutting RPP to the intersected surface to determine which abutting RPP the ray is entering. Variable M is set to a negative one whenever DO loop index I is odd; M is set to a positive one when index I is even. For negative M, the left abutting RPP in the packed word is tested. For positive M, the right abutting RPP in the packed word is tested.

The statement

```
      IF(M.GT.0)GOTO 50
```

is used to test variable M to determine which RPP in the packed abutting RPP word is to be tested.

The statements

```
CALL UN2(LOCAT,I1,I2)
LOCAT=LOCAT+1
IRP=I1
GOTO 70
```

executed if M is negative, are used to retrieve the next packed abutting RPP word. The left abutting RPP in the packed word is equated to argument IRP, and the program branches to determine if the RPP is an abutting RPP. LOCAT is indexed to the next abutting RPP word for unpacking when needed.

The statement

```
50 IRP=I2
```

executed when M is positive, is used to set the right abutting RPP in the packed word equal to argument IRP for subsequent testing.

The statement

```
70 LS=(1-LSURF)/2
```

is used to compute a control variable based on the intersected surface number in order to insure that identically numbered sides of the intersected RPP and the potential abutting RPP are never compared since these boundaries would never be together.

The statements

```
   DO 80 J=1,3
   IF(J.EQ.LS)GOTO 80
   IF((S(IRP,2*J-1)-XP(J))*(XP(J)-S(IRP,2*J)).LT.0.)GOTO 90
80 CONTINUE
   RETURN
```

consist of a DO loop which is used to determine if the intersect occurs within the boundaries of the potential abutting RPP by evaluating, assuming an intersect on an X plane,

$$\begin{aligned} Ymin &< Y_c < Ymax \\ Zmin &< Zc < Zmax \end{aligned}$$

where Yc and Zc are the y-z coordinates of the intersect and Ymin, Ymax, Zmin, and Zmax are the bounding plane coordinates of the potential abutting RPP.

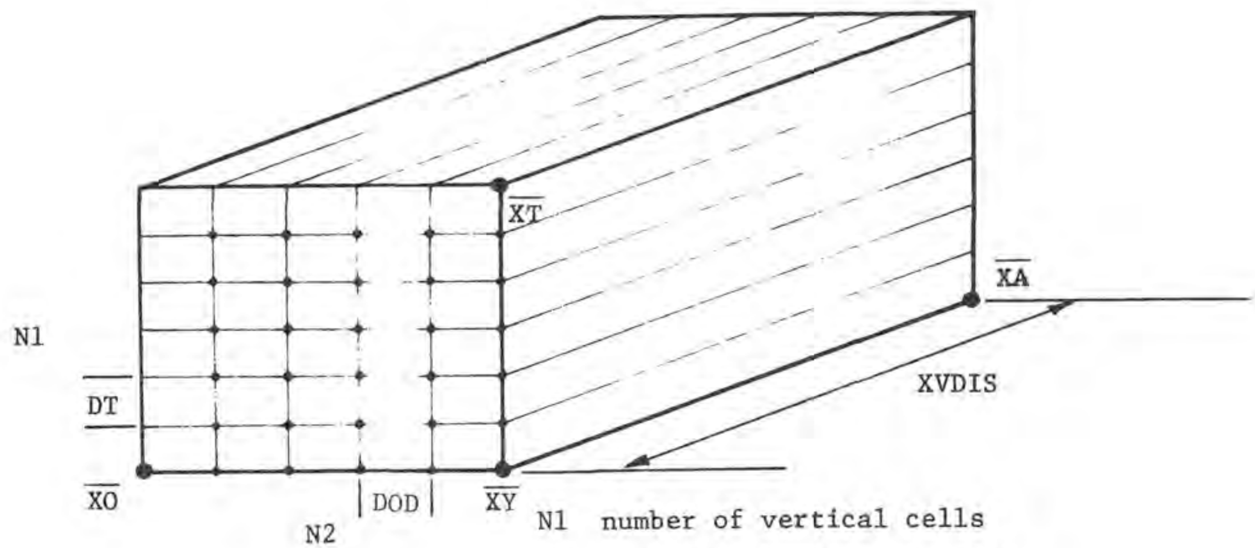
The statements

```
   90 CONTINUE
      IRP=0
      RETURN
      END
C
C
```

are used to continue to the next potential abutting RPP if the intersect did not occur within the boundaries of the present potential RPP. If the intersect occurs on the face of an RPP that has no abutting RPP, IRP is set to zero, and control is returned to Subroutine G1.

Subroutine VOLUM

Subroutine VOLUM is called by the MAIN program if the option variable for calling this subroutine is set to one. The purpose of this routine is to compute the volumes of each region within a given imaginary box, where the box is defined as in Figure 73.



N1 number of vertical cells

N2 number of horizontal cells

$\overline{XV}$ lower right corner of box front

$\overline{XT}$ upper right corner of box front

$\overline{XO}$ lower left corner of box front

$\overline{XA}$ lower right corner of box end

DT vertical dimension of cell

DOD horizontal dimension of cell

XVDIS distance from front to back of box

FIG. 73. Volume Geometry

Rays are traced from the lower right corner of each cell from the front to the back of the box and the distances through each region are stored in an array. When all rays have been traced and the total distance through each region accumulated, the volumes are computed from the region distances and the cell dimensions.

The statement

```
C
C     COMPUTE VOLUMES BY REGION IN VOLUME DEFINED BY BOX
C
      DIMENSION VASTER(1000),WAB(3),WTB(3),WOB(3),DSP(3),
     1  XV(3),XT(3),XA(3),XO(3),XP(3),XTEMP(3)
```

is used to dimension arrays for use by this subroutine.

The statements

```
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/WALT/LIRFO,NG1ERR
```

are used to pass information into and out of this subroutine.

The statements

```
C
  901 FORMAT(3E20.8)
  902 FORMAT(2E20.8)
  903 FORMAT(1H0,10X,6HVERTEX,14X,6HTOP.PT,14X,6HBOT.PT,14X,7HSIDE.PT)
  904 FORMAT(4E20.8)
  905 FORMAT(1H0, 8X,12HDELTA ON TOP,E20.8,10X,10HSIDE DELTA,E20.8)
  906 FORMAT(2I10)
  908 FORMAT(1H0, 2X,18HSTARTING REGION IS,I5)
  909 FORMAT(1H0,16HVASTER OVERWRITE,5X,6HNRMAX=,I5)
  910 FORMAT(I10,E20.8)
  911 FORMAT(1H0,8HBAD CARD/I10,E20.8,14H NOT PROCESSED)
  912 FORMAT(I10,E20.8,5X,E20.8,5X,E9.2)
  913 FORMAT(1H0,5HSUMV=,5X,E20.8)
```

are used to format data for input and output.

The statements

```
C
      READ (5,906)IR,NG1ERR
      IF(NG1ERR.LE.0)NG1ERR=25
```

are used to enter the region number of starting point $\overline{XV}$ and to enter the allowable number of Subroutine G1 errors. If the number of allowable errors entered is less than or equal to zero, the allowable errors in Subroutine G1 are set to 25.

The statements

```
C
C     ENTER COORDINATES OF BOX
C
      READ (5,901)(XV(I),I=1,3)
      READ (5,901)(XT(I),I=1,3)
      READ (5,901)(XO(I),I=1,3)
      READ (5,901)(XA(I),I=1,3)
C
C     ENTER CELL SIZE
C
      READ (5,902)DOD,DT
      WRITE (6,903)
      WRITE (6,904)(XV(J),XT(J),XO(J),XA(J),J=1,3)
      WRITE (6,905)DOD,DT
      WRITE (6,908)IR
```

are used to enter and print out the coordinates of the four corners that define the box; to enter and print out the cell dimensions; and to print out the starting region number.

The statement

```
      IF(NRMAX.GT.2000)WRITE (6,909)NRMAX
```

is used to determine if the number of regions used to describe the target geometry is excessive relative to the array size used in this subroutine. If it is, an error message with the number of regions is printed out.

The statements

```
      CALL DCOSP(XV,XT,WTB)
      CALL DCOSP(XV,XO,WOB)
      CALL DCOSP(XV,XA,WAB)
```

are used to compute the direction cosines of the vectors from the vertex point to each of the three other points of the box.

The statement

```
XVDIS=XDIST(XV,XA)
```

is used to compute the length of the box from front to back by the use of Function XDIST.

The statements

```
      TESTON=0.
      TESTOV=0.
      XTEMP(1)=0.
      DO 10 I=1,NRMAX
      VASTER(I)=0.
   10 CONTINUE
```

are used to zero subroutine variables and the array for storing the distances through the regions.

The statements

```
      JIR=IR
      IRJ=IR
```

are used to save the starting region number for later use in the subroutine.

The statements

```
C
C*    COMPUTE NUMBER OF HORIZONTAL AND VERTICAL CELLS
C
      N2=XDIST(XV,XO)/DOD+1.
      N1=XDIST(XV,XT)/DT+1.
```

are used to compute the number of horizontal and vertical cells from the dimensions of the cell and the size of the front plane of the box.

The statement

```
C
C5     TRACE RAYS FROM LOWER RIGHT CORNER OF EACH CELL
C
       DO 300 J=1,N2
```

is used to begin a DO loop which will trace a ray from the lower right corner of each cell to the end of the box, accumulating distances that the rays travel through each region. This DO statement indexes columns of cells on the face of the box.

The statements

```
       DO 100 I=1,3
       DSP(I)=WTB(I)*DT
       XB(I)=XV(I)
       WB(I)=WAB(I)
  100  CONTINUE
```

consist of a DO loop which is used to compute the vector from the present origin to the next origin in the current column from where the next ray is to be traced. The coordinates of the origin and the direction cosines of the current ray are then assigned.

The statements

```
       S1=0.
       IR=JIR
```

are used to initialize variable S1, the distance to the next region, and to initialize to the region number of the origin of the current ray.

The statements

```
C
C6     TRACE ALL RAYS FROM COLUMN OF CELLS
C
       DO 200 I=1,N1
       NASC=-1
```

are used to begin a DO loop for tracing each ray from each cell in the present column of cells. For each new ray, NASC is set to a -1 to indicate to Subroutine G1 that a new ray is to be traced.

The statements

```
C
C7     TRACE RAY THROUGH BOX VIA SUBROUTINE G1
C
  110 CALL G1(S1,IRPRIM,XP)
      IF(IERR.GE.NG1ERR)GOTO 400
      VASTER(IR)=VASTER(IR)+S1
      IF(DIST.GE.XVDIS)GOTO 115
      IF(IRPRIM.LE.0)GOTO 120
      IR=IRPRIM
      GOTO 110
```

are used to call Subroutine G1 to trace the given ray to the end of the box. Subroutine G1 returns when a new region has been encountered and the number of errors that occurred in Subroutine G1 is compared to the allowable limit. If the allowable number of errors was exceeded, the program is terminated. If the allowable error limit was not exceeded, the distance through the given region is added to the previous distance that the ray(s) has travelled through the region. A test is made to determine if the new region was encountered beyond the end of the box. If so, the routine branches to subtract that distance beyond the end of the box. A test is made to determine if the new region is an RPP boundary; if so, the ray is terminated. If the new region was encountered before the end of the box, the present region is updated to the new region, and the program branches to call Subroutine G1 again to continue the ray.

The statement

```
  115 VASTER(IR)=VASTER(IR)-(DIST-XVDIS)
```

which is executed when the new region encountered lies beyond the end of the box, is used to subtract that portion of the distance beyond the end of the box.

The statements

```
  120 XTEMP(1)=WB(1)
      XTEMP(2)=WB(2)
      XTEMP(3)=WB(3)
      IR=JIR
```

are used to save the ray direction cosines and to initialize to the starting region number of the present ray.

The statements

```
TESTON=TESTON-DT
IF(TESTON.GT.0.)GOTO 180
```

are used to determine if the origin of the next ray in the current column is in the same region as the origin of the ray just completed.

The statements

```
WB(1)=WTB(1)
WB(2)=WTB(2)
WB(3)=WTB(3)
NASC=-1
```

are used to assign direction cosines of the vector from point $\overline{XV}$ to $\overline{XT}$, and to initialize for a new ray to determine the region number of the next origin.

The statements

```
C
C8     DETERMINE REGION OF NEXT ORIGIN OF RAY IN COLUMN
C
       CALL G1(S1,IRPRIM,XP)
       IF(IERR.GE.NG1ERR)GOTO 400
       IF(S1-DT)130,160,170
```

are used to call Subroutine G1 to find the distance to the next region in the direction of the next origin. If an excessive number of errors did not occur in Subroutine G1, the distance to the next region is compared with the distance to the next origin.

The statements

```
   130 IR=IRPRIM
       JIR=IR
       CALL G1(S1,IRPRIM,XP)
       IF(IERR.GE.NG1ERR)GOTO 400
       IF(DIST-DT)140,160,170
```

are executed if the distance to the next origin is greater than the distance to the next region. These statements update to the new region number and save it in case the next origin is in that region. If an excessive number of errors did not occur in Subroutine G1, the distance to the next region is compared with the distance to the next origin to determine if the origin is in the new region returned by Subroutine G1.

The statements

```
140 IF(IRPRIM)150,210,130
150 STOP
```

are executed if the distance to the next origin is again greater than the distance to the next region. If this next region number is negative, the program stops because this is an error. If the region number is zero, the RPP enclosing the target geometry has been intersected and the program branches to index to the next column of cells. If the region number is positive, the program branches to find the next region where the origin could be located.

The statements

```
160 IR=IRPRIM
    JIR=IR
```

are executed if the next origin and a new region occur simultaneously. These statements update to the new region and store it in a save area for later use.

The statement

```
170 TESTON=S1
```

which is executed if the distance to the next region is greater than the distance to the next origin, is used in the program to determine if succeeding ray origins are in the same region.

The statements

```
C
C10   SHIFT ORIGIN OF RAY TO NEXT CELL IN COLUMN
C
  180 DO 190 JI=1,3
      WB(JI)=XTEMP(JI)
      XB(JI)=XB(JI)+DSP(JI)
  190 CONTINUE
  200 CONTINUE
```

consist of a DO loop which is used to compute the coordinates of the origin of the next ray and to restore the original direction cosines of the ray. The program then loops to trace the next ray.

The statement

```
C
C11   ONE COLUMN OF CELLS COMPLETE - SHIFT TO NEXT COLUMN
C
  210 NASC=-1
```

is used to set NASC to a -1 to indicate to Subroutine G1 that a new ray is to be traced. This statement is executed when all of the rays in one column have been traced, and the rays in the next column are now to be traced.

The statements

```
      DO 220 I=1,3
      WB(I)=WOB(I)
      XB(I)=XV(I)
  220 CONTINUE
```

consist of a DO loop which is used to assign the direction cosines and origin of a ray to be traced from the first origin of the current column to the first origin of the next column.

The statements

```
      JIR=IRJ
      IR=JIR
      TESTDN=0.
      TESTOV=TESTOV-DOD
      IF(TESTOV)230,230,280
```

are used to initialize to the region number of the first origin of the current column. TESTDN is initialized to zero for later use in determining the regions of the origins of the next column. The distance to the next region in the direction of the first origin of the next column is computed to determine if the first origin of the next column is in the same region as the first origin of the current column.

The statements

```
C
C12   DETERMINE REGION OF FIRST ORIGIN OF NEXT COLUMN
C
  230 CALL G1(S1,IRPRIM,XP)
      IF(IERR.GE.NG1ERR)GOTO 400
      IF(S1-DOD)240,260,270
```

are executed if the region number of the first origin of the next column is not known. These statements are used to compute the distance to the next region in the direction of the first origin of the next column by calling Subroutine G1. This distance is compared with the distance between the columns after checking for more-than-allowable errors.

The statements

```
  240 IR=IRPRIM
      IRJ=IR
      CALL G1(S1,IRPRIM,XP)
      IF(IERR.GE.NG1ERR)GOTO 400
      IF(DIST-DOD)250,260,270
```

are executed if the distance to the first origin of the next column is greater than the distance to the next region. These statements update to the new region number and save it in case the origin is in this region. Subroutine G1 is again called to determine the distance to the following region; Subroutine G1 compares the total distance with the distance to the origin, after checking for more-than-allowable errors.

The statements

```
250 IF(IRPRIM)255,400,230
255 STOP
```

are executed if the distance to the first origin of the next column is again greater than the distance to the next region. If this next region number is negative, the program stops since this is an error. If the region number is zero, the RPP enclosing the target geometry has been intersected, and the program branches to index to the next column. If the region number is positive, the program branches to find another region where the next origin could be located.

The statements

```
260 IR=IRPRIM
    IRJ=IR
```

are executed if the first origin of the next column and an intersect with a new region occur simultaneously. These statements update to the new region and save it for later use.

The statement

```
270 TESTOV=S1
```

which is executed if the distance to the next region is greater than the distance to the next origin, is used in the program to determine if succeeding ray origins are in the same region as previous ray origins.

The statements

```
C
C13   SHIFT ORIGIN OF NEXT RAY TO FIRST ORIGIN OF NEXT COLUMN OF CELLS
C
  280 DO 290 I=1,3
      XA(I)=XA(I)+WOB(I)*DOD
      XV(I)=XV(I)+WOB(I)*DOD
      XT(I)=XT(I)+WOB(I)*DOD
  290 CONTINUE
```

consist of a DO loop which is used to shift the vertex, top point and the box end point to the next column of cells.

The statements

```
      JIR=IR
  300 CONTINUE
```

are used to save the region number of the first origin of the new column and to branch to trace the rays of the new column.

The statements

```
C
C14   ALL RAY DISTANCES THROUGH EACH REGION IN BOX ACCUMULATED
C
  400 READ (5,910)IR1,VR
      IF(IERR.GE.NG1ERR)GOTO 500
      IF(IR1.LE.0)IR1=NRMAX+1
      SUMV=0.
```

are executed when all rays have been traced; when more than the allowable limit of errors occurred in Subroutine G1; or when the front plane of the box intersects an RPP boundary. These statements are used to enter a region number and its pre-computed volume. The Subroutine G1 error counter is compared with the allowable limit of errors; if more than the allowable limit of errors occurred, the program branches to return control to the MAIN program. If less than the number of allowable errors occurred in Subroutine G1, and no region number was entered, the region number is equated to one greater than the number of regions in the target geometry. SUMV, the variable for summing the total volume of all regions within the box, is also set to zero.

The statement

```
C
C15   COMPUTE VOLUME OF EACH REGION IN BOX
C
      DO 450 I=1,NRMAX
```

is used to begin a DO loop which will compute and print out the volume of each region encountered within the volume of the box and to compute the percent error with the pre-computed volumes if the pre-computed volume was entered.

The statement

```
VASTER(I)=VASTER(I)*DOD*DT
```

is used to compute the volume of each region within the volume of the box.

The statement

```
IF(I-IR1)410,430,420
```

is used to determine if the current region of the DO loop has a pre-computed volume entered for computing percent error.

The statements

```
410 WRITE (6,910)I,VASTER(I)
    GOTO 440
```

are used to write out the region number and the computed volume of the region if the entered region number is greater than the current region number of the DO loop.

The statements

```
420 WRITE (6,911)IR1,VR
    READ (5,910)IR1,VR
    GOTO 410
```

are executed if the region number of the DO loop is greater than the last region number entered. These statements are used to print out the entered region number and pre-computed volume and to enter the next card with region number and pre-computed volume.

The statements

```
C
C16   COMPUTE PERCENT ERROR FOR PRE-COMPUTED VOLUME OF GIVEN REGION
C
  430 XPERC=100.*(VASTER(I)/VR-1.)
      WRITE (6,912)I,VASTER(I),VR,XPERC
```

```
      VASTER(I)=VR
      READ (5,910)IR1,VR
```

are executed when the entered region is equal to the region number of the DO loop. These statements are used to compute the percent error between the computed volume and the pre-computed volume of the current region. The region number, computed volume, pre-computed volume, and percent error are printed, and the next region number with pre-computed volume is entered.

The statements

```
C
C17   COMPUTE TOTAL VOLUME OF ALL REGIONS WITHIN BOX
C
  440 SUMV=SUMV+VASTER(I)
  450 CONTINUE
```

are used to accumulate the total volume of the regions considered and to transfer to consider the next region.

The statements

```
      WRITE (6,913)SUMV
  500 IERR=0
      RETURN
      END
C
C
```

are used to print out the total of the volume of all of the regions encountered within the box. The Subroutine G1 error counter is set to zero, and this subroutine returns control to the MAIN program.

Subroutine AREA

Subroutine AREA is called by the MAIN program if the option variable for calling this subroutine is set to one. The purpose of this subroutine is to compute the presented area of an object or target as viewed from any azimuth and elevation angle.

The statement

```
DIMENSION XP(3),WP(3),XBS(3),CONVRT(4,4),TYPEUN(4)
```

is used to dimension arrays for use in this subroutine.

The statements

```
COMMON ASTER(10000)
COMMON/PAREM/XB(3),WB(3),IR
 COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
COMMON/CAL/NIR,SLOS,ANGLE,NTYPE,SSPACE,L,XS(3),WS(3),
1  TRAVEL,SN,V,H,IVIH
COMMON/WALT/LIRFO,NG1ERR
COMMON/CELL/CELSIZ
COMMON/ENGEOM/LEGEOM
```

are used to pass information into and out of this subroutine.

The statements

```
C
 901 FORMAT(7I10,6X,2A2)
 902 FORMAT(6E12.5)
 908 FORMAT(1H0,22HMEMORY OVERLAP IN AREA,5X,7HLEGEOM=,I6,
    1     5X,6HLAREA=,I6,5X,6HLIRFO=,I6)
 909 FORMAT(1H0,13HERROR IN AREA,5X,9HICODE = 0)
 910 FORMAT(1H0,8HAZIMUTH=,F10.3,5X,10HELEVATION=,F10.3)
 911 FORMAT(1H0,12HCELL SIZE IS,F4.1,1X,1HX,F4.1,1X,A2,1H.,10X,
    1     12HAREAS IN SQ.,1X,A2,1H.)
 912 FORMAT(1H0,5HICODE,19X,4HAREA/)
 913 FORMAT(I5,15X,F12.5)
 914 FORMAT(1H0,15HPRESENTED AREA=,F12.5)
 915 FORMAT(1H0,18HNUMBER OF CELLS IS,I5,10X,
    1     22HNUMBER OF CELLS HIT IS,I5)
```

are used to format data for input and output and to format output messages.

The statements

```
C
      DATA HHIN,HHFT,HHCM,HHMB,HHBB/2HIN,2HFT,2HCM,2HM ,2H  /
      TYPEUN(1)=HHIN
      TYPEUN(2)=HHFT
      TYPEUN(3)=HHCM
      TYPEUN(4)=HHMB
```

are used to enter Hollerith constants data. These constants, which are codes for inches, feet, centimeters, and meters, are equated to the four elements of array TYPEUN.

The statements

```
      CONVRT(1,1)=1.
      CONVRT(1,2)=.00694444444444444
      CONVRT(1,3)=6.451625806
      CONVRT(1,4)=.0006451625806
      CONVRT(2,1)=144.
      CONVRT(2,2)=1.
      CONVRT(2,3)=929.0341161
      CONVRT(2,4)=.09290341161
      CONVRT(3,1)=.15499969
      CONVRT(3,2)=.001076386736
      CONVRT(3,3)=1.
      CONVRT(3,4)=.0001
      CONVRT(4,1)=1549.9969
      CONVRT(4,2)=10.7636736
      CONVRT(4,3)=10000.
      CONVRT(4,4)=1.
```

are used to assign constants for converting the area from one unit of measurement to another. The constants for array CONVRT are used for converting an area as follows:

CONVRT(1,1)	square inches to square inches
CONVRT(1,2)	square inches to square feet
CONVRT(1,3)	square inches to square centimeters
CONVRT(1,4)	square inches to square meters
CONVRT(2,1)	square feet to square inches
CONVRT(2,2)	square feet to square feet
CONVRT(2,3)	square feet to square centimeters
CONVRT(2,4)	square feet to square meters
CONVRT(3,1)	square centimeters to square inches
CONVRT(3,2)	square centimeters to square feet
CONVRT(3,3)	square centimeters to square centimeters
CONVRT(3,4)	square centimeters to square meters

CONVRT(4,1)	square meters to square inches
CONVRT(4,2)	square meters to square feet
CONVRT(4,3)	square meters to square centimeters
CONVRT(4,4)	square meters to square meters

The statement

```
BLANK=HHBB
```

is used to equate two Hollerith blanks to variable BLANK for determining if there are blanks in card input fields.

The statements

```
C
C1      COMPUTE AND INITIALIZE AREA FOR STORING PRESENTED AREA
C       BY COMPONENT CODE
C
        LAREA=LIRFO-1000
        IF(LAREA.GE.LEGEOM)GOTO 10
```

are used to compute a starting location in the ASTER array for storing the presented area of the target by component code (1-999). The location is computed to be 1000 words before location LIRFO, the storage area for the identification code and component code data. A test is made to verify that this new storage area does not overlap the end of the target geometry data.

The statements

```
WRITE (6,908)LEGEOM,LAREA,LIRFO
STOP
```

are executed if the new storage area overlaps the end of the target geometry data in the ASTER array. These statements cause an error message to be printed out with the pertinent subroutine variables and to terminate the program.

The statements

```
10 LAREA1=LIRFO-1
   DO 20 L=LAREA,LAREA1
   ASTER(L)=0.
20 CONTINUE
```

are used to compute the last word of the array for storing the presented area of the target by component code and to zero the entire array with the DO loop.

The statements

```
C
C2    READ GRID INPUT PARAMETERS
C
      READ (5,901)NX,NY,IRSTRT ,IENC,NG1ERR,NSTART,NEND,CELLUN,AREAUN
      READ (5,902)A,E,ENGTH,ZSHIFT,GROUND
      READ (5,902)XSHIFT,YSHIFT,CELSIZ
```

are used to enter the grid input parameters.

The statements

```
C
C3    INITIALIZE PARAMETERS NOT SET BY INPUT
C
      IF(IRSTRT .LE.0)IRSTRT=1
      IF(CELSIZ .LE.0.)CELSIZ=4.
      IF(NSTART.LE.0)NSTART=1
      IF(NG1ERR.LE.0)NG1ERR=25
      IF(AREAUN.EQ.BLANK)AREAUN=HHIN
      IF(CELLUN.EQ.BLANK)CELLUN=HHIN
```

are used to determine or initialize the starting region number, the cell size, the cell number from which ray tracing is to begin, the limit of allowable errors that can occur in Subroutine G1, the measurement units for expressing the area, and the measurement units of the cell.

The statements

```
C
C4    DETERMINE MEASUREMENT UNITS AND COMPUTE GRID CELL AREA
C
      DO 30 I=1,4
      IF(CELLUN.EQ.TYPEUN(I))GOTO 40
   30 CONTINUE
   40 DO 50 J=1,4
      IF(AREAUN.EQ.TYPEUN(J))GOTO 60
   50 CONTINUE
   60 AREAC=CELSIZ*CELSIZ*CONVRT(I,J)
```

are used to determine the measurement units of the cell and the desired measurement units to describe the area. The area of the cell is computed and converted to the selected measurement units (inches^2, feet^2, centimeters^2, or meters^2).

The statements

```
C
      RADIAN=.017453292519943
      AR=A*RADIAN
      ER=E*RADIAN
      SA=SIN(AR)
      CA=COS(AR)
      SE=SIN(ER)
      CE=COS(ER)
```

are used to compute the sine and cosine of the azimuth and elevation angles from the azimuth and elevation angles that were entered in degrees.

The statements

```
      KL=NX*NY
      NHIT=0
```

are used to compute the number of cells in the grid plane and to initialize the number of hits counter, NHITS, to zero.

The statement

```
C
CS    PROCESS KL CELLS IN GRID PLANE
C
      DO 200 KK=NSTART,KL
```

is used to begin a DO loop which will construct a grid plane, fire a ray from each cell in the grid plane, and compute and store the presented area by component code when the ray intersects the target.

The statements

```
      WB(1)=-CE*CA
      WB(2)=-CE*SA
      WB(3)=-SE
```

are used to compute the direction cosines for the ray normal to the grid plane and directed toward the target geometry.

The statements

```
C
C4     COMPUTE ROW AND COLUMN NUMBER OF GRID CELL
C
       II=((KK-1)/NX)+1
       J=KK-(II-1)*NX
```

are used to compute the row and column of a specific grid square from which the ray is to be fired toward the target geometry.

The statements

```
C
       CELL2=.5*CELSIZ
       V=FLOAT((NY/2)-II)*CELSIZ +CELL2
       VREF=V+CELL2
       H=FLOAT((NX/2)- J)*CELSIZ +CELL2
       HREF=H+CELL2
```

are used to locate the lower left corner of the grid square. V represents the vertical distance from the center of the grid plane and H represents the horizontal distance. VREF and HREF refer to the center of the current grid square.

The statements

```
       IV=RAN(-1)*10.
       IH=RAN(-1)*10.
       IVIH=10*IH+IV
```

are used to compute two random numbers between zero and nine.

The statements

```
C
C7     COMPUTE RANDOM POINT WITHIN GRID CELL
C
       V=V+CELSIZ *FLOAT(IV)/10.+CELSIZ /20.
       H=H+CELSIZ *FLOAT(IH)/10.+CELSIZ /20.
```

are used to locate one random point out of a possible 100 random points within the current grid cell.

The statements

```
C
      XBS(1)=XSHIFT-V*CA*SE-H*SA
      XBS(2)=YSHIFT-V*SA*SE+H*CA
      XBS(3)=ZSHIFT+V*CE
```

are used to transform the point within the current grid cell to the coordinate system of the target and, at the same time, to effectively move the grid plane and target system coordinate origins to a new location specified by the variables XSHIFT, YSHIFT, and ZSHIFT.

The statement

```
      CALL TROPIC(WP)
```

is used to call Subroutine TROPIC which generates random direction cosines from an isotropic distribution.

The statements

```
      XBS(1)=XBS(1)+WP(1)*1.0E-4
      XBS(2)=XBS(2)+WP(2)*1.0E-4
      XBS(3)=XBS(3)+WP(3)*1.0E-4
```

are used to move the point within the current grid cell by a very small amount in a random direction.

The statements

```
C
CB    CONVERT GRID PLANE COORDINATES TO COORDINATES OF TARGET
C
      XB(1)=XBS(1)-ENGTH*WB(1)
      XB(2)=XBS(2)-ENGTH*WB(2)
      XB(3)=XBS(3)-ENGTH*WB(3)
```

are used to back the point out of the target by an amount ENGTH, which was entered with the input data. The ray will originate from this point and pass through the position of the point before it was moved to its present position.

The statement

```
      IF(XB(3).LE.GROUND)GOTO 200
```

is used to determine the position of the origin of the ray. If the origin occurs below ground level, the ray will not be fired, and the program branches to compute the origin of the next ray.

The statements

```
C
C9     TRACE RAY TO FIRST TARGET COMPONENT HIT
C
       IR=IRSTRT
       NASC=-1
  110  CALL G1(S1,IRPRIM,XP)
```

are used to initialize the region identifier to the starting region of the ray and to initialize the variable NASC to -1 to indicate to Subroutine G1 that a new ray is being fired. Subroutine G1 is called to move the point on the ray to the next region.

The statements

```
       IF(IERR.GE.NG1ERR)RETURN
       IF(IRPRIM.LT.0)GOTO 200
       IF(NASC.LE.NRPP)IRPRIM=0
       IF(IRPRIM.EQ.0)GOTO 200
```

are used to determine if more than the allowable number of errors occurred in Subroutine G1. If not, further tests are made to determine if an RPP boundary has been intersected. If the intersect occurs at an RPP boundary, the ray has missed the target and the program branches to process the next ray.

The statements

```
       LOC=LIRFO+IRPRIM-1
       CALL UN2(LOC,ICODE,IDENT)
```

are used to locate and unpack the identification code and the component code from the ASTER array for the region returned by Subroutine G1.

The statements

```
       IDENT=IDENT-1
       IF(IDENT-(IDENT/10)*10.EQ.0)GOTO 120
       IR=IRPRIM
       GOTO 110
```

are used to determine if the material of the intersected region is part of the target. If not, the region number is updated, and control is returned to again call Subroutine G1 to return the next region encountered along the ray. One is subtracted from variable IDENT, since a one was added before IDENT was packed to prevent packing a negative number.

The statements

```
120 IF(ICODE.NE.0)GOTO 130
    WRITE (6,909)
    GOTO 200
```

are used to test the component code if the identification code test revealed that the target was intersected. If the component code is zero, an error message is printed out, and the program continues to process the next ray.

The statements

```
130 LOC=LAREA*ICODE-1
    ASTER(LOC)=ASTER(LOC)+AREAC
    NHIT=NHIT+1
200 CONTINUE
```

are executed if both the identification code and component code agree that the target has been hit by the current ray. These statements compute a storage location in the ASTER array and add the presented area of the target indexed by the component code of the material hit. The number of hits counter is incremented by one for the current ray, and the program branches to process the next ray.

The statements

```
C
C10   PRINT RESULTS
C
      WRITE (6,910)A,E
      WRITE (6,911)CELSIZ, CELSIZ, CELLUN,AREAUN
      WRITE (6,912)
      SUMA=0.
```

are executed when all of the rays of the grid have been processed. These statements are used to print out the azimuth and elevation angles of the grid plane, the dimensions of the cells, and the measurement units of the cell and of the computed areas. Column headings for printout of the presented areas by component codes are printed out, and the storage location for summing all of the presented area is initialized to zero.

The statements

```
      DO 250 I=1,999
      LOC=LAREA+I-1
      IF(ASTER(LOC).EQ.0.)GOTO 250
      WRITE (6,913)I,ASTER(LOC)
      SUMA=SUMA+ASTER(LOC)
  250 CONTINUE
```

consist of a DO loop which is used to index each of the elements in the array for storing presented areas by component code (1-999). If there is no presented area for a given component code, the loop indexes to the next component code location. If there is a presented area for a given component code, the component code and its presented area are printed out. The presented area is added to location SUMA to obtain a total presented area.

The statements

```
      WRITE (6,914)SUMA
      WRITE (6,915)KL,NHIT
      RETURN
      END
C
C
```

are executed when all of the presented areas by component code have been printed out and all of the presented areas have been summed. These statements print out the total presented area, the number of rays, and the number of rays that hit the target; and return control to the MAIN program.

Subroutine TESTG

Subroutine TESTG is called by the MAIN program if the option variable for calling Subroutine TESTG is set to one. The purpose of this routine is to trace a given number of rays between different sets of two points when given the coordinates and the region number of the points. Outputs of this routine consist primarily of leaving region, entering region, distance the ray has travelled into a region, coordinates of the point on the ray, and total distance the ray has travelled.

The statement

```
C
C1    TRACE A RAY BETWEEN TWO GIVEN POINTS XB TO XBF
C
      DIMENSION XP(3),XBF(3)
```

is used to dimension two three-element arrays for the coordinates of the beginning and ending points.

The statements

```
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/WALT/LIRFO,NG1ERR
```

are used to pass information into and out of this subroutine via COMMON statements.

The statements

```
C
  901 FORMAT(2I10)
  902 FORMAT(1H0,22HNUMBER OF SPECIAL RAYS,I5)
  903 FORMAT(3E15.7,3I15)
  904 FORMAT(1H0,5HSTART,5X,4H XB=,3E15.7,8H IRSTRT=,I5/
     1      4H END,7X,4HXBF=,3E15.7,8H  IRFIN=,I5)
  905 FORMAT(1H0,3HWB=,3E15.7,5X,6HRANGE=,E15.7)
  906 FORMAT(1H0,8X,2HIR,4X,6HIRPRIM,12X,2HS1,13X,2HXP,13X,2HYP,
     1      13X,2HZP,12X,4HDIST)
  907 FORMAT(2I10,5X,5E15.7)
  908 FORMAT(1H0,21HTROUBLE IN REGION IR=,I10)
```

are used to format data for input and output and to format output messages.

The statements

```
C
C2    ENTER NUMBER OF RAYS
C
      READ (5,901)NRAYS,NG1ERR
      WRITE (6,902)NRAYS
      IF(NG1ERR.LE.0)NG1ERR=25
```

are used to enter and print out the number of different rays to be traced and to enter the number of allowable errors for Subroutine G1. If the number of allowable G1 errors entered is equal to or less than zero, it is set to 25 allowable Subroutine G1 errors.

The statements

```
C
C3    TRACE GIVEN NUMBER OF RAYS
C
      DO 50 IRAY=1,NRAYS
```

are used to begin a DO loop which will enter and print out the data for each ray to be traced between two given points.

The statements

```
C
C4    ENTER POINT COORDINATES AND REGION OF EACH
C
      READ (5,903)XB,IRSTRT
      READ (5,903)XBF,IRFIN
      WRITE (6,904)XB,IRSTRT,XBF,IRFIN
```

are used to enter and print out the coordinates of the beginning and ending points of the ray and the region number where each point is located.

The statements

```
      RANGE=XDIST(XB,XBF)
      CALL DCOSP(XB,XBF,WB)
      WRITE (6,905)WB,RANGE
```

are used to compute the distance between the two points using Function XDIST, to compute the direction cosines of the line between the two points using Subroutine DCOSP, and to print out the distance and the direction of the line between the two points.

The statements

```
IR=IRSTRT
NASC=-1
WRITE (6,906)
```

are used to initialize variable IR to the starting region number and to set variable NASC to a -1 to indicate to Subroutine G1 that a new ray is to be fired. The table headings for the output are then printed out.

The statements

```
C
C     TRACE RAY TO NEXT REGION INTERSECT
C
   10 CALL G1(S1,IRPRIM,XP)
      IF(IERR.GE.NG1ERR)GOTO 60
      WRITE (6,907)IR,IRPRIM,S1,XP,DIST
```

are used to call Subroutine G1 to determine the distance to the next region, the number of the next region, the coordinates at the intersect of the next region, and the total distance the current ray has travelled from the first point. If more than the allowable number of errors occurred in Subroutine G1, control is transferred to zero the error counter and return to the MAIN program. If less than the allowable number of errors occurred in Subroutine G1, the present region number, the entering region number, the distance to the entering region, the coordinates of the intersect at the new region, and the total distance from the first point to the new region are printed out under the applicable column headings for this intersect.

The statements

```
IF(DIST.GE.RANGE)GOTO 30
IF(IRPRIM.LE.0)GOTO 20
IR=IRPRIM
GOTO 10
```

are used to determine if the distance that the ray has travelled is greater than or equal to the distance between the two points, or if the new region is outside the enclosing geometry of the enclosing RPP. If not, the current region number is updated to the new region, and control is branched to again call Subroutine G1 to continue the ray.

The statements

```
C
   20 WRITE (6,908)IR
      GOTO 50
```

are executed if the new region is outside the enclosing RPP of the target geometry. These statements print out the old region number with an error message and then branch to start the next ray (if any).

The statements

```
   30 IF(IR.NE.IRFIN)GOTO 20
   50 CONTINUE
```

are executed if the distance travelled is greater than or equal to the distance between the two points. These statements determine if the old region is the same as the region of the end point. If not, the old region number, with an error message, is printed out, and the program continues with the next ray (if any). If the regions are the same, the program continues with the next ray (if any).

The statements

```
   60 IERR=0
      RETURN
      END
C
C
```

are executed when all rays have been processed or when the allowable number of errors in Subroutine G1 was exceeded during the processing of one of the rays. These statements set the error counter to zero and return control to the MAIN program.

LIST OF SYMBOLS AND ABBREVIATIONS (SIMULATION MODEL)

Definitions of variable names utilized in this analysis program are contained in the following list of symbols and abbreviations (simulation model).

The variable names are presented in the following groups:

1. COMMON Statements

   Variable names appearing in the COMMON statements of all routines are listed in one group.

2. SUBROUTINES

   Variable names not appearing in COMMON statements are listed by subroutine.

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

COMMON

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
ANGLE	α	Angle between the normal and the ray at the intersect	Degrees
CA	$\cos \alpha$	Cosine of the azimuth angle of the ray with respect to the target geometry origin	ND
CE	$\cos \theta$	Cosine of the elevation angle of the ray with respect to the target geometry origin	ND
CELSIZ	CELSIZ	Length and width of each cell in the grid plane	Inches
D1	---	Distance from the first intersect on the target to the center plane of the target	Inches
D2	---	Distance from the last intersect on the target to the center plane of the target	Inches
DIST	---	Distance from the start of a new region through the region until a new region is encountered	Inches
H	H	Horizontal distance from center of grid plane to random point in specified grid cell	Inches
HREF	H_{ref}	Horizontal distance from the center of the grid plane to the center of the grid square	Inches
I15	---	Value 2^{15}. Used for packing and unpacking data in a single word	ND
I30	---	Value 2^{30}. Used for packing and unpacking data in a single word	ND
IA(9)	---	Array for entering the logical operator or when entering region data during Subroutine GENI	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

COMMON (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IC(4)	---	Array for entering alphameric program control data and the abbreviation of the body type during Subroutine GENI	ND
IENC	---	Region number enclosing the target and attack plane	ND
IENTLV	---	Option variable used to determine if Subroutine GENI is to print out the region enter/leave tables	ND
IERR	---	Variable used to count the number of errors in the geometry input. Also used to count the number of errors in Subroutine G1	ND
IERRO	---	Counter for the number of 0 component code errors	ND
IGRID	---	Grid square of the origin of the current ray	ND
IN(9)	---	Array for entering the operator (+ or -) and the body number when entering region data during Subroutine GENI	ND
INORM	---	Control variable for Subroutine ARS to either compute the normal distance (INORM=1), or to compute the line-of-sight distance (INORM=0)	ND
IR	---	Region number where the point along the ray is presently located	ND
IRANDM	---	Control variable passed from MAIN to Function RAN for computing a random number	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

COMMON (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IRAYSK	---	Control variable passed from MAIN to Subroutine GRID to skip a random number of cells in the grid plane if IRAYSK is not equal to zero	ND
IRN	---	Region number of region type data entered	ND
IRSTRT	---	Region number of ray origin	ND
IT(10)	---	Title of the problem of up to 60 alphanumeric characters	ND
ITAPE8	---	Option variable for suppressing printout (ITAPE8=0)	ND
ITESTG	---	Option variable used to determine if Subroutine TESTG is to be called by the MAIN program	ND
ITR(200)	---	Storage array for recording ray contact data: surface number, body number, next region number	ND
IVIH	---	Two-digit random number computed in Subroutine GRID for printout by Subroutine TRACK	ND
IVOLUM	---	Option variable used to determine if Subroutine VOLUM is to be called by the MAIN program	ND
IWOT	---	Option variable used to determine if Subrouting GRID data is to be written on output tape 1	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

COMMON (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IYES	---	Integer variable equal to one which is used for option testing	ND
KLOOP	---	Internal ray counter for keeping track of each new ray in Subroutine G1	ND
KHIT	---	Counter for keeping track of the number of components hit along a given ray	ND
L	---	Counter for the number of intersects along a given ray	ND
LABUT	---	Location of the beginning abutting RPP data in the MASTER-ASTER array	ND
LBASE	---	Beginning location of the MASTER-ASTER array (usually one)	ND
LBODY	---	Location of the body pointers in the MASTER-ASTER array	ND
LDATA	---	Temporary address of data in the MASTER-ASTER array	ND
LEGEOM	---	Location of the end of the geometry data processed by Subroutine GENI	ND
LIO	---	Location of the beginning of a temporary storage area in the MASTER-ASTER array for use by Subroutine G1	ND
LIRFO	---	Starting location of the region ID data in the MASTER array	ND
LMAX	---	Total number of intersects that occur along a given ray	ND
LOCDA	---	Location of data in the MASTER array	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

COMMON (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
LOOP	---	Value of KLOOP when ray data was stored	ND
LREGD	---	Beginning location in the MASTER-ASTER array of the region data pointer/number of bodies in region	ND
LRI	---	Surface number of the entering intersect	ND
LRIN	---	Beginning location in the ASTER array of the RIN data	ND
LRO	---	Surface number of the exit intersect	ND
LROT	---	Beginning location in the ASTER array of the ROUT data	ND
LRPPD	---	Beginning location in the ASTER array of the RPP minimum/maximum values	ND
LSCAL	---	Beginning location in the ASTER array where the scalar data is to be entered by Subroutine GENI	ND
LSURF	---	Surface number of body surface hit (negative if exit intersect)	ND
LTRIP	---	Beginning location in the ASTER array where the triplet data is to be entered by Subroutine GENI	ND
NASC	---	Current body number (-1 means start new ray, -2 means find normal distance)	ND
NBODY	---	Number of bodies used to describe the target geometry other than RPP's	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

COMMON (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
NDQ	---	Upper limit or size in words of the MASTER-ASTER array, usually 10,000	ND
NG1ERR	---	Maximum number of errors that are allowed in Subroutine G1	ND
NIR	---	Region identification (region component code)	ND
NN	---	Variable made up of the body number plus the number of RPP's	ND
NO	---	Integer variable equal to zero which is used for option testing	ND
NRMAX	---	Total number of regions used to describe the target geometry	ND
NRPP	---	Number of rectangular parallele-pipeds used to enclose the target geometry	ND
NTYPE	---	Space code of the region following the next intersect	ND
PINF	---	Value 10^{50}. Used to represent infinity	ND
RIN	RIN	Distance along the ray from the beginning of a given region to the entry intersect of the body under consideration	Inches
ROUT	ROUT	Distance along the ray from the beginning of a given region to the exit intersect of the body under consideration	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

COMMON (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
SA	$\sin \alpha$	Sine of the azimuth angle of the ray with respect to the target geometry	ND
SE	$\sin \theta$	Sine of the elevation angle of the ray with respect to the target geometry	ND
SLOS	---	Line-of-sight distance through region following present intersect	Inches
SN	---	Normal distance through a given region	Inches
SSPACE	---	Line-of-sight distance through space following a given intersect	Inches
TR(200)	---	Storage array for recording ray line-of-sight distance from contact to contact	ND
TRAVEL	---	Line-of-sight distance from origin of ray to present intersect	Inches
V	V	Vertical distance from center of grid plane to random point in specified grid cell	Inches
VREF	V_{ref}	Vertical distance from center of grid plane to center of grid square	Inches
WB(3)	$\overline{WB}$	Present direction cosines of the ray	ND
WS(3)	$\overline{WS}$	Original direction cosines of the ray	ND
X(6)	---	Temporary storage array for entering the six bounding planes of an RPP	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

COMMON (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
XB(3)	$\overline{XB}$	x, y, and z coordinates of the origin of the ray with respect to a given intersect	Inches
XBS(3)	$\overline{X}_p$	x, y, and z coordinates of a point in a plane through the center of the target geometry through which the ray will pass	Inches
XS(3)	---	x, y, and z coordinates of the origin of the ray in the grid plane	Inches

LIST OF SYMBOLS AND ABBREVIATIONS (SIMULATION MODEL)

SUBROUTINE MAIN

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A	---	Six-element array for entering a description of the region type data	ND
I	---	Index in DO loops or for entering and writing out data	ND
ICODE	---	Item code of component	ND
IDENT	---	Space code and special identification of region	ND
IRDTP4	---	Option variable for entering the target geometry data from tape	ND
IRN	---	Area for entering region numbers when entering region data	ND
IWRTP4	---	Option variable for writing out the target geometry data onto tape	ND
K	---	Location pointer to the MASTER array for storing ICODE and IDENT region data	ND
NAREA	---	Number of aspect angles to be processed by Subroutine AREA	ND
NOAA	---	Number of aspect angles to be processed by Subroutine GRID	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GENI

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A	---	Distance from the center to one end of an ellipsoid of revolution	Inches
ASQ	---	Distance squared from center to one end of an ellipsoid of revolution	Inches^2
C	---	Distance from the center to a focus of an ellipsoid of revolution	Inches
CX	---	x component distance from the center to a focus of an ellipsoid of revolution	Inches
CY	---	y component distance from the center to a focus of an ellipsoid of revolution	Inches
CZ	---	z component distance from the center of a focus of an ellipsoid of revolution	Inches
FX(20)	---	Temporary array used to enter body triplet and scalar data. Also used for manipulating and computing additional data before storing the data into the MASTER-ASTER array in its final format	ND
HDN	---	Dot product of the height vector and the normal to the base ellipse of a truncated elliptic cone	Inches
I	---	Index for various DO loops	ND
II	---	Pointer to the first position of a group of data in the ASTER array	ND
I2	---	Pointer to the last position of a group of data in the ASTER array	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GENI (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IAA(1)	---	Storage location containing the Hollerith logical operator "bbb"	ND
IAA(2)	---	Storage location containing the Hollerith logical operator "OBb"	ND
IAA(3)	---	Storage location containing the Hollerith logical operator "bRb"	ND

(Table continued on next page)

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GENI (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IAA(4)	---	Storage location containing the Hollerith logical operator "Rbb"	ND
IAA(5)	---	Storage location containing the Hollerith logical operator "RAb"	ND
IAA(6)	---	Storage location containing the Hollerith logical operator "ARb"	ND
IAA(7)	---	Storage location containing the Hollerith logical operator "bAb"	ND
IAA(8)	---	Storage location containing the Hollerith logical operator "Abb"	ND
IAN(1)	---	Storage location containing the integer 4 for converting the Hollerith logical operator to a numerical value of four	ND
IAN(2)	---	Storage location containing the integer 1 for converting the Hollerith logical operator to a numerical value of one	ND
IAN(3)	---	Storage location containing the integer 1 for converting the Hollerith logical operator to a numerical value of one	ND
IAN(4)	---	Storage location containing the integer 1 for converting the Hollerith logical operator to a numerical value of one	ND
IAN(5)	---	Storage location containing the integer 2 for converting the Hollerith logical operator to a numerical value of two	ND
IAN(6)	---	Storage location containing the integer 2 for converting the Hollerith logical operator to a numerical value of two	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GENI (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IAN(7)	---	Storage location containing the integer 3 for converting the Hollerith logical operator to a numerical value of three	ND
IAN(8)	---	Storage location containing the integer 3 for converting the Hollerith logical operator to a numerical value of three	ND
IBL	---	Storage location containing one Hollerith blank for testing for blank fields on card input	ND
II	---	Number of bodies in the larger of two regions that is used as the upper limit of a DO loop for comparing each item in the smaller region with each item in the larger region	ND
IK	---	Lower limit of a DO loop used to print out the MASTER-ASTER array three words at a time where IK is the location of the first of the three words	ND
IK2	---	Upper limit of a DO loop used to print out the MASTER-ASTER array three words at a time where IK2 is the location of the third of three words	ND
IO	---	Number of bodies in the smaller of two regions that is used as the upper limit of a DO loop for comparing each item in the smaller region with each item in the larger region	ND
IOP	---	Logical operator of the body to be tested or compared from the enter/leave tables in the ASTER array	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GENI (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IOPER	---	Refer to IOP	ND
IOPI	---	Refer to IOP	ND
IOPO	---	Refer to IOP	ND
IPRIN	---	Option control variable for printing out the entire MASTER-ASTER array	ND
IR	---	Number of the region entered from the card input	ND
IRCHEK	---	Option control variable for verifying the validity of the region enter/leave tables	ND
IS	---	Value of +1 or -1 used in preparing the enter/leave tables. For IS = -1 the leave table is prepared and for IS = +1 the enter table is prepared	ND
ITEMP	---	Variable used to represent the location of a region data pointer word	ND
ITY(1)	---	Storage location containing the Hollerith string BOX to represent a box	ND
ITY(2)	---	Storage location containing the Hollerith string SPH to represent a sphere	ND
ITY(3)	---	Storage location containing the Hollerith string RCC to represent a right circular cylinder	ND
ITY(4)	---	Storage location containing the Hollerith string REC to represent a right elliptic cylinder	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GENI (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
ITY(5)	---	Storage location containing the Hollerith string TRC to represent a right truncated cone	ND
ITY(6)	---	Storage location containing the Hollerith string ELL to represent an ellipsoid of revolution	ND
ITY(7)	---	Storage location containing the Hollerith string RAW to represent a right angle wedge	ND
ITY(8)	---	Storage location containing the Hollerith string ARB to represent an arbitrary polyhedron	ND
ITY(9)	---	Storage location containing the Hollerith string TEC to represent a truncated elliptic cone	ND
ITY(10)	---	Storage locating containing the Hollerith string TOR to represent a torus	ND
ITY(11)	---	Storage location containing the Hollerith string ARS to represent an arbitrary surface	ND
ITYPE	---	Variable first used to store the string of the body type when a card is entered. It is later used to store the integer equivalent of the body type	ND
IWH	---	Pointer to the location of pointer data in the MASTER array	ND
J	---	Index used to represent region numbers	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GENI (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
J1-J12	---	Variables used to represent the first and last location of either the region enter table or region leave table during the printout of these tables	ND
JJ	---	Variable used as an index when comparing the region data of two different regions	ND
K	---	Index for use in DO loops and in entering data	ND
KI	---	Index for the number of bodies in the larger of two regions where the region tables are being verified	ND
KLK	---	Variable used to represent the location of a region data word	ND
KO	---	Index for the number of bodies in the smaller of two regions where the region tables are being verified	ND
KRI	---	Variable used to represent the location of a region data pointer word	ND
KRJ	---	Variable used to represent the location of a region data pointer word	ND
L	---	Temporary storage location or coordinate index	ND
L1	---	Variable used to represent the location of the last word in the enter/leave table	ND
LAR	---	Variable used to represent the last location of RPP data in the MASTER-ASTER array	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GENI (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
LBOD	---	Variable used to represent the beginning location of the body data pointers	ND
LBOT	---	Variable initially used to represent the beginning location of the scalar data. This value changes as additional body data is stored backward from the beginning location of the scalar data	ND
LD	---	Variable used to represent the location of the last word of the body data pointers	ND
LE	---	Variable used to represent the number of data elements or pointers for a given body	ND
LEAV	---	Pointer to the first location of the region leave table	ND
LEGEOM	---	Pointer to the last location of the geometry data processed by Subroutine GENI	ND
LENLV	---	Variable used to represent the beginning location of the leave/ enter tables	ND
LENT	---	Pointer to the first location of the region enter table	ND
LL	---	Variable used as a counter to represent the number of errors in the region data when the checking option is performed	ND
LOC	---	Variable used to represent the location of a specific word in the MASTER-ASTER array	ND
LOCI	---	Variable used to represent the location of the region data for a specific region	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GENI (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
LOCJ	---	Variable used to represent the location of the region data for a specific region	ND
LREGL	---	Variable used to represent the beginning location of the region data	ND
LS1	---	Variable used to indicate whether triplet data (LS1=0) or scalar data (LS1=1) is to be stored by Subroutine SEE3	ND
LSUB	---	Variable used to represent the number of unused words between the last pointer word and the first body data word	ND
LT	---	Pointer to a reference location for storing pointers to triplet data	ND
M	---	Counter or pointer for the region data in the body pointer section of the MASTER array	ND
MIS	---	Counter for the number of matching operator/body combinations in another region	ND
MM	---	Variable used to index to specific body data words in the region data	ND
MMM	---	Index used to prepare either the region leaving table or region entering table	ND
N	---	Index used to represent body number or region number	ND
NBI	---	Region table body number to be compared	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GENI (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
NBNR	---	Variable that represents the total number of geometric shapes used to describe the target geometry. It is used in printing the enter/leave table	ND
NBO	---	Region table body number under test	ND
NBOD(I)	---	Eleven-element array used to count the number of times each of the eleven body shapes was used to describe the target geometry	ND
NC	---	Number of bodies in a given region	ND
NEAV	---	Number of bodies in a given region leave table	ND
NENT	---	Number of bodies in a given region enter table	ND
NO1(M) NO2(M)	---	Two three-element arrays used to represent the last 15 bits and the 15 bits previous to the last 15 bits respectively of a computer word during the MASTER-ASTER array printout option when three words are printed out at a time	ND
NOO(M)	---	Three-element array used to represent the three locations of the words for printout during the MASTER-ASTER array printout option	ND
NUM	---	Body number candidate for a region enter or leave table	ND
NUMI	---	Number of bodies in the smaller region when checking validity region data	ND
NUMJ	---	Number of bodies in the larger region when checking validity of region data	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GENI (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
O4(M)	---	Three element array used to store the three words to be printed out during the MASTER-ASTER array printout option	ND
TT(3)	---	Three element array used to store the x, y, and z coordinates of the normal to the base ellipse of a TEC	Inches
TT1(3)	---	Three element array used to store the x, y, and z coordinates of the semi-major axis of the base ellipse of a TEC	Inches
TT2(3)	---	Three element array used to store the x, y, and z coordinates of the semi-minor axis of the base ellipse of a TEC	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE RPPIN

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
I	---	Location of the pointers for a given side of an RPP, or Number of an RPP for use in DO loops	ND
I2 I5	---	Number of abutting RPP's in packed format	ND
I3 I6	---	Pointer to the boundary coordinate for a given side of an RPP	ND
II	---	Pointer to the location of the boundary coordinate for a given side of an RPP, or Number of an RPP for use as the lower limit in a DO loop	ND
J	---	Index used to represent the side or number of an RPP	ND
JJ	---	Location of the pointers for a given side of an RPP	ND
K KK	---	Location of the pointers for a given side of an RPP, or Index used to represent side pairs of an RPP for use in a DO loop	ND
K2	---	Even numbered side of a given RPP	ND
K21	---	Odd numbered side of a given RPP	ND
K41	---	Number representing the sum of the numbers of two opposite sides of an RPP	ND
L	---	Index for referencing storage in the LRPPD section	ND
LAR	---	Location of the last word of RPP data	ND

LIST OF SYMBOLS AND ABBREVIATIONS
SUBROUTINE RPPIN (Concluded) (SIMULATION MODEL)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
LAST	---	Location of the last word of RPP data	ND
LL	---	Counter for the number of abutting RPP's for a given side of an RPP	ND
M	---	Switch used in determining in which part of a packed word the number of an abutting RPP is to be stored	ND
N	---	Represents the number of an RPP or the number of a side of an RPP	ND
NC	---	Represents the side of an RPP opposite the side under test	ND
NN	---	Sum of the numbers representing two opposite sides of an RPP	ND
NRPP1	---	One less than the number of RPP's used to represent the target geometry	ND
X(J)	---	Value of a boundary coordinate for a side of a given RPP	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ALBERT

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A	A	Coefficient of x in the equation of the plane	ND
A2B2C2	---	Sum of the squares of the x, y, and z components of the equation of the plane	Inches^2
AA(I,J)	---	24-element two-dimensional array for entering and storing the coordinates of the eight vertices of the ARB	Inches
B	B	Coefficient of y in the equation of the plane	ND
C	C	Coefficient of z in the equation of the plane	ND
D	D	Constant term in the equation of the plane	ND
D12	---	Square of the length between two vertices of a side of the ARB	Inches^2
D1210	---	D12 times 10^{-12}	Inches^2
D2	---	Perpendicular distance from the fourth vertex to the plane formed by the first three vertices	Inches
D22	---	$(D2)^2$	Inches^2
F(4)	---	Four-element array for storing the results of the four vertices not part of the plane under consideration when substituted into the plane equation	ND
FX	---	Six-element array that contains the coordinates of the first two vertices of the ARB	Inches
I	---	Index for entering the coordinates of the vertices of the ARB	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ALBERT (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IA	---	24-element two-dimensional array for entering and storing the four ordinal vertex numbers for six planes of the ARB	ND
IC	---	Fourth ordinal number of one of the vertices of a plane	ND
IWH	---	Pointer returned by Subroutine SEE3 to the location where the ARB data was stored	ND
IX	---	First ordinal number of one of the vertices of a plane	ND
IY	---	Second ordinal number of one of the vertices of a plane	ND
IZ	---	Third ordinal number of one of the vertices of a plane	ND
J	---	Index in a DO loop for storing coordinates of two vertices. Also used to count the number of vertices on the positive side of the plane under test	ND
K	---	Index for referencing elements in array FX. Also used as an index to zero array F	ND
L	---	Index for storing data into the F array	ND
LBOT	---	Pointer to the location where data was last stored by Subroutine SEE3	ND
LS1	---	Variable used to indicate to Subroutine SEE3 whether triplet data (LS1=0) or scalar data (LS1=1) is to be stored	ND
M	---	Variable for counting the number of vertices on the negative side of the plane under test	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ALBERT (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
N	---	Variable for counting the number of vertices remaining on the plane under test	ND
NDQ	---	Upper limit pointer of the MASTER-ASTER array	ND
S	---	Square root of the sum of the squares of the x, y, and z coefficients	Inches
WX	---	x direction cosine of a vector from the origin perpendicular to the plane	Inches
WY	---	y direction cosine of a vector from the origin perpendicular to the plane	Inches
WZ	---	z direction cosine of a vector from the origin perpendicular to the plane	Inches
X1	---	x coordinate of the first vertex of a plane	Inches
X2	---	x coordinate of the second vertex of a plane	Inches
X3	---	x coordinate of the third vertex of a plane	Inches
X4	---	x coordinate of the fourth vertex of a plane	Inches
Y1	---	y coordinate of the first vertex of a plane	Inches
Y2	---	y coordinate of the second vertex of a plane	Inches
Y3	---	y coordinate of the third vertex of a plane	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ALBERT (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
Y4	---	y coordinate of the fourth vertex of a plane	Inches
Z1	---	z coordinate of the first vertex of a plane	Inches
Z2	---	z coordinate of the second vertex of a plane	Inches
Z3	---	z coordinate of the third vertex of a plane	Inches
Z4	---	z coordinate of the fourth vertex of a plane	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ARIN

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
I	---	DO loop index equal to curve number during input of ARS point data	ND*
L	---	DO loop index for referencing storage locations in the ARS point data section	ND
LBOT	---	Pointer to beginning location of storage area for ARS	ND
LDATA	---	Pointer to next available location in body pointer section of MASTER array	ND
LOC	---	Storage location pointer for ARS point data section	ND
LOCC	---	Storage location pointer for ARS point data section	ND
L1	---	DO loops lower limit for referencing storage locations in ARS point data section	ND
L2	---	DO loops upper limit for referencing storage locations in ARS point data section	ND
M	---	Number of curves used to describe given ARS	ND
MN	---	Total number of points used to describe given ARS	ND
N	---	Number of points per curve	ND
NP	---	Number of points to be stored; NP=2*N*(M-1); points stored in pairs between consecutive curves	ND
NSTR	---	Total number of storage words required for the given ARS	ND
NT	---	Number of triangles described	ND

*Non-dimensional

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ARIN (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
NT	---	Number of non-degenerate triangles in given ARS	ND
UW(3)	---	x, y, z coordinates of vector between first point and second point of given triangle of ARS	ND
VW(3)	---	x, y, z coordinates of vector between first point and third point of given triangle of ARS	ND
W(3)	---	x, y, z coordinates of first point of given triangle of ARS	ND
WN(3)	---	x, y, z coordinates of vector formed from cross product of vectors $\overline{UW}(3)$ and $\overline{VW}(3)$	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE SEE3

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
FX	---	x coordinate of the triplet data, or the value of the scalar quantity	ND
FXX	---	y coordinate of the triplet data, or the value of the scalar quantity	ND
FXXX	---	z coordinate of the triplet data, or the value of the scalar quantity	ND
I	---	Index for searching through the triplet/scalar data	ND
IWH	---	Pointer to location of the triplet or scalar data	ND
LBOT	---	Beginning locations of the triplet/scalar data section	ND
LDATA	---	Pointer to the next available location in the body section of the MASTER array	ND
LS1	---	Variable used to indicate whether triplet data (LS1=0) or scalar data (LS1=1) is in the argument list	ND
NDQ	---	Last location of the MASTER-ASTER array	ND
NDQ2	---	NDQ-2	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GRID

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A	A	Azimuth angle	Degrees
AR	α	Azimuth angle	Radians
CELL2	D/2	One-half the total number of grid cells	ND
E	E	Elevation angle	Degrees
ENGTH	ENGTH	Back-off distance of the shifted point of the given cell	Inches
ER	θ	Elevation angle	Radians
GROUND	---	z-coordinate of ground level	Inches
ICENTR	---	Control variable for originating the ray from the center of the cell when equal to one	ND
IH	I_h	Random number for computing a random horizontal point within a given cell	ND
II	I	Variable that represents the row number of the grid	ND
IV	I_v	Random number for computing a random vertical point within a given cell	ND
J	J	Variable that represents the column number of the grid	ND
KK	k	Index of the major DO loop that represents the cell number	ND
KK1	---	Index that represents an x, y, or z coordinate in a DO loop	ND
MSHIFT	---	Random number between 0 and 24 that determines the random number of cells to be skipped	ND
NEND	N	Number of the last cell in the grid	ND
NSTART	---	Starting cell number, usually the first cell in the grid	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE GRID (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
NX	N_x	Number of horizontal cells in the grid plane	ND
NY	N_y	Number of vertical cells in the grid plane	ND
RADIAN	R	One degree in radians = 0.017453292519943	Radians
WP(3)	---	x, y, z coordinates of random direction cosines returned by Subroutine TROPIC	Inches
XSHIFT	XSHIFT	Distance target origin and grid plane center is effectively shifted in the X direction	Inches
YSHIFT	YSHIFT	Distance target origin and grid plane center is effectively shifted in the Y direction	Inches
ZSHIFT	ZSHIFT	Distance target origin and grid plane center is effectively shifted in the Z direction	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TRACK

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
ANGLE1	---	Save area for obliquity angle of intersect for first half of output line	ND
ERROR	---	Two-element array containing Hollerith data for possible printout	ND
D1	---	Distance from the first intersect of the target to the center plane of the target	Inches
D2	---	Distance from the last intersect of the target to the center plane of the target	Inches
I	---	Index in a DO loop for referencing the TR and ITR arrays. Also used as an index for the x, y, and z coordinates in another DO loop	ND
I12	---	Value for packing data into array ITR. $I12 = 2^{12} = 4096$	ND
IDENT		Region identification space code	ND
IH	---	Horizontal cell number from center of grid	ND
IRPRIM	---	Region number returned by Subroutine G1	ND
IV	---	Vertical cell number from center of grid	ND
JCNT	---	Counter used to count the spaces the ray encounters in the target	ND
JERRO	---	Index for ERROR array. Set to 2 if 0 component code error occurs.	ND
KLSURF	---	Surface number where the ray intersects the body (negative for exit intersect)	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TRACK (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
LOC	---	Location of the region identity code in the region type data section	ND
MARMR	---	Flag for indicating armor material	ND
MSKRT	---	Flag for indicating skirt material	ND
MTARG	---	Flag for indicating target	ND
MVOL	---	Flag for indicating interior volume	ND
N	---	Number of components hit by ray	ND
NIR1	---	Save area for region identification (vehicle component) of intersect for first half of output line	ND
NTYPE1	---	Save area for the type of space following region of intersect for first half of output line	ND
S1	---	Distance to the next region returned by Subroutine GI	Inches
SLOS1	---	Save area for the line-of-sight distance through region for first half of output line	ND
SN1	---	Save area for normal distance through region for first half of output line	ND
SPACE1	---	Save area for line-of-sight distance through space for first half of output line	ND
SUM	---	Summing location for computing distance from first target intersect to center plane of target	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TRACK (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
XP	---	x, y, and z coordinates of the new position of the ray returned by Subroutine G1	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE CALC

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A	A	Variable used in the ELL section to represent the length of the major axis	Inches
A1	a	Variable used in the REC section to represent the length from the center along the major axis	Inches
A2	b	Variable used in the REC section to represent the length from the center along the minor axis	Inches
ASQ	$(R1/R2)^2$ BSQ	Variable used in the TEC section to represent the square of the radius of the intersection ellipse along the semi-major axis	Inches^2
BSQ	$[\gamma \cdot R4+R2(1-\gamma)]^2$	Variable used in the TEC section to represent the square of the radius of the intersection ellipse along the semi-minor axis	Inches^2
C	c	Variable used in the REC and TEC section to represent the distance from the center of the ellipse to the foci	Inches
DIS	---	Distance from ray origin to intersect in ARS section	Inches
DIV	$\sqrt{A^2+B^2+C^2}$	Variable used in the ARB section to represent the square root of the sum of the squares of the x, y, and z coefficients of the equation of the intersected plane	Inches
GAMMA	$\frac{(\overline{X}-\overline{V})\cdot\overline{N}}{\overline{H}\cdot\overline{N}}$	Variable used in the TEC section to represent the ratio of the height of the hit along the normal to the distance between the two planar surfaces	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE CALC (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
HDN	$\overline{H} \cdot \overline{N}$	Variable used in the TEC section to represent the length of the height vector when projected onto the normal to the base ellipse	Inches
HF(3)	---	Array used in TEC section for the coordinates of the height vector	Inches
HH	$(\overline{X}-\overline{V}) \cdot \overline{N}$	Variable used in the TEC section to represent the length of the vector from the vertex to the intersect when projected onto the normal to the base ellipse	Inches
I	---	Variable used throughout the program as an index	ND
IDENT	---	Variable used to represent the space code of a particular region	ND
IEMP	---	Temporary storage for entering the coordinate data for the box	ND
IJK	---	Variable used throughout the program as an index for retrieving data from the ASTER array	ND
IJK1	---	Refer to IJK	ND
IJK2	---	Refer to IJK	
IJK3	---	Refer to IJK	ND
ISPOT	---	Variable used as an index to locate specific region data in the ASTER array	ND
ITYPE	---	Variable used to represent the body type of the intersected body	ND
J	---	Variable used throughout the program as an index	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE CALC (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
K	---	Variable used in the RAW section as an index to retrieve data from the ASTER array	ND
KCOM	---	Variable used in the BOX section to test the surface number of the intersect for odd or even status	ND
L1	---	Pointer to the location of the x, y, and z coefficients for the equation of the intersected plane of the ARB	ND
LA	---	Pointer to the location of the direction cosines of the semi-major axis of the base ellipse of the TEC	ND
LH	---	Variable used in the BOX section as an index to locate the three $\bar{H}$ vectors of the BOX in the ASTER array	ND
LK	---	Refer to K	ND
LKK	---	Variable used in the RPP section as an index to locate data in the ASTER array	ND
LN	---	Pointer to the location of the direction cosines of the normal for the TOR and TEC	ND
LOC	---	Location of the pointers in the LBODY section for the body being tested	ND
LOCARS	---	Beginning location of intersect data for the ARS	ND
LR1	---	Variable used in the REC, ELL, REC, TEC, and TOR sections as an index to locate data in the ASTER array	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE CALC (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
LR2	---	Variable used in the REC, ELL, and TEC sections as an index to locate data in the ASTER array	ND
LR3	---	Variable used in the TEC section as an index to locate data in the ASTER array	ND
LS	---	Variable used in the ELL section as an index to locate the length of the major axis from the ASTER array	ND
LSPT	---	Variable used in the ARB section to represent the location of the intersected plane data in the ASTER array	ND
LSURF	---	Surface number of the body where the intersect occurs (negative if an exit intersect)	ND
LV	---	Variable used as an index to locate the coordinates of the vertex in the ASTER array	ND
LV1	---	Variable used as an index to locate the coordinates of the vertex or height vector in the ASTER array	ND
LV2	---	Variable used as an index to locate the coordinates of the height vector in the ASTER array	ND
LV3	---	Refer to LV1	ND
M	---	Variable used as an index to locate data in the ASTER array	ND
MK	---	Variable used in the BOX section as an index to locate data in the ASTER array	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE CALC (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
NBO	---	Solid number of the body under test	ND
NEXREG	---	Region number of the region following the next intersect	ND
R1	R1	Variable used in section TEC to represent the length of the major radius of the base ellipse, and in section TOR to represent the major radius	Inches
R2	R2	Variable used in section TEC to represent the length of the minor radius of the base ellipse	Inches
R4	R4	Variable used in section TEC to represent the length of the minor radius of the top ellipse	Inches
S1	---	Normal distance through the region that is returned by Subroutine G1	Inches
SUM	---	Variable used in sections REC and RAW to compute the dot product of two vectors, and in section ARB to compute the sum of the squares of the x, y, and z coefficients of the intersected plane	Inches^2
TAU	$(R1/R2)^2$	Variable used in section TEC to represent the square of the ratio of the semi-major axis radius to the semi-minor axis radius of the base ellipse	ND
TEM(3)	---	Three-element array used to store the x, y, and z coordinates of a vector	Inches
TEM1(3)	---	Refer to TEM(3)	Inches
TEMP(3)	---	Refer to TEM(3)	Inches
TEMP1(3)	---	Refer to TEM(3)	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE CALC (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
TLK	n	Variable used in the RAW section to compute and represent the scalar length of a vector normal to the slanted side of the RAW	Inches
TWOA	2a	Variable used in the TEC section to represent the length of the intersection ellipse along the semi-major axis	Inches
VF(3)	---	Array for storing the vertex coordinates of the TEC	Inches
WI(3)	$\overline{WI}$	Coordinates of the direction cosines of the height vector of the RCC	Inches
WN(3)	$\overline{WN}$	Three-element array used to store the x, y, and z coordinates of the direction cosines of a unit vector	Inches
XI(3)	$\overline{X}$	Three-element array used to store the x, y, and z coordinates of the intersect point	Inches
XMID(3)	$(\overline{H1}-\overline{H2})$	Three-element array used in the RAW section to represent the x, y, and z coordinates of the H1-H2 vector	Inches
XNOS	Q	Variable used to represent a constant multiplier of the direction cosines of a ray and has a value of +1 or -1 and is used to direct the normal to a surface into the body for an entry intersect and away from the body for an exit intersect	ND
XP(3)	$\overline{XH}$	Three-element array used in the RCC section to represent the x, y, and z coordinates of the intersect point projected onto the height vector	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE G1

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
I	---	Index for referencing the temporary working storage section of Subroutine G1 (LIO)	ND
I1	---	Entering surface number of an intersect for a given body from LIO	ND
I2	---	Exit surface number of an intersect for a given body from LIO	ND
I3	---	Equivalent of LOOP for a given body from LIO. Not used in Subroutine G1	ND
ICODE	---	Item code of the region before the present intersect	ND
ICODE1	---	Item code of the region after the present intersect	ND
IDENT	---	Space code of the region before the present intersect	ND
IDENT1	---	Space code of the region after the present intersect	ND
IH	---	Horizontal grid cell of the ray (from the center cell)	ND
IJK	---	Pointer for storing or locating data in the MASTER-ASTER array	ND
IRP	---	Number of an abutting RPP	ND
IRPRIM	---	Number of the next region	ND
ITEMP	---	Pointer to data in the MASTER-ASTER array	ND
ITY	---	Body type number (1-12) of the current body	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

G1 (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
ITYPE	---	Unpacked body type number (1-11) of the current body	ND
IV	---	Vertical grid cell of the ray (from the center cell)	ND
J	---	Pointer to the possible region being entered by the ray	ND
J1	---	Pointer to the first region in the enter (or leave) table for the current body	ND
J2	---	Pointer to the last region in the enter (or leave) table for the current body	ND
LEAV	---	Pointer to the region leaving table for the current body	ND
LENT	---	Pointer to the region entering table for the current body	ND
LION	---	Pointer to the last location in LIO, the temporary working storage for Subroutine G1	ND
LOC	---	Pointer to data in the MASTER-ASTER array	ND
LSURT	---	Surface number for the current intersect of the current body (negative if an exit intersect)	ND
LTRUE	---	Indicator (0 or 1) to Subroutine G1 from Subroutine WOWI if point $\overline{XP}$ is in the region passed to Subroutine WOWI	ND
NASCT	---	Body number of the current intersect	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

G1 (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
NBO	---	Body number of the body under test	ND
NC	---	Number of bodies in the region description	ND
NEAV	---	Number of regions in the region entering table	ND
NHIT	---	Count of the number of bodies hit at a single intersect	ND
S1	---	Distance the ray has travelled into the region	Inches
SM	---	Distance to the next intersect of the body being tested that is greater than the distance travelled thus far	Inches
XBD(3)	---	x,y, and z coordinates of the ray when an error in Subroutine G1 occurred	Inches
XP(3)	---	x,y, and z coordinates of the position of the point on the ray	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE WOWI

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IJK	---	Pointer for locating and storing data in Subroutine G1 working storage, LIO	ND
IOP	---	Body operator of the body when unpacked from the region data section	ND
IOPER	---	Body operator of the body when unpacked from the region data section	ND
ITEMP	---	Pointer for locating body data pointers for the body being tested	ND
ITY	---	Body type number (1-12) of the current body	ND
LOCD	---	Pointer to the packed operator and body number for a specific body in the region under test	ND
LTRUE	---	Indicator (0 or 1) to Subroutine G1 if point $\overline{XB}$ is within the region	ND
N	---	Region body sequence number of the body being tested	ND
NBO	---	Body number of the body under test	ND
NC	---	Number of bodies in the region description	ND
NN	---	Region body sequence number of the body being compared	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE RPP

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
I	---	DO loop index for representing one of six sides of the RPP	ND
II	i	Index for denoting x, y, and z plane pairs	ND
J	---	Index that represents the x, y, or z coordinate of a point	ND
L	---	Index counter for the number of sides intersected	ND
LR(L)	---	Surface number of intersect point	
LST(1)	1	Integer that represents X plane	ND
LST(2)	1	Integer that represents X plane	ND
LST(3)	2	Integer that represents Y plane	ND
LST(4)	2	Integer that represents Y plane	ND
LST(5)	3	Integer that represents Z plane	ND
LST(6)	3	Integer that represents Z plane	ND
NBO	---	Body number of the body under test	ND
PR(L)	---	Location in subroutine for RIN and ROUT	Inches
TEMP	$\overline{XS}_I - \overline{XB}_J$	Numerator of the equation for distance to the intersect point of ray with plane	Inches
TRY	$\overline{XS}_I - \overline{XB}_S / \overline{WB}_J$	Distance from ray origin to intersect point with plane	Inches
XRY	$\overline{XB} + \overline{WB} \cdot S_I$	Intersect coordinate on a plane	Inches
XS(I)	$\overline{XS}_I$	Boundary coordinate for a given plane	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE BOX

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A	---	Dot product $\overline{H1} \cdot \overline{H1}$	Inches^2
CM	RINi	Computed value for distance to the first intersection with the box, RIN	Inches
CP	ROUTi	Computed value for distance to the second intersection with the box, ROUT	Inches
I	---	DO loop index for computing the face pair	ND
IH1	---	Location in ASTER array which contains $\overline{H1}$ coordinates	ND
IH2	---	Location in ASTER array which contains $\overline{H2}$ coordinates	ND
IH3	---	Location in ASTER array which contains $\overline{H3}$ coordinates	ND
II	---	Integer used to compute the face number of the box	ND
IV	---	Location in the ASTER array which contains the vertex coordinates	ND
J	---	Index for looping through the x, y, and z coordinates for computing the equation for the distance to the intersects	ND
JA	---	x, y, or z coordinate of the $\overline{H1}$ vector	Inches
JV	---	x, y, or z coordinate of the vertex	Inches
LI	---	Face number for ray's first intercept	ND
LO	---	Face number for ray's second intercept	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE BOX (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
LOC	---	Index used to retrieve pointers to the $\overline{H1}$ and $\overline{H2}$ coordinates	ND
VP	$(\overline{V}-\overline{XB}) \cdot \overline{Hi}$	Dot product $(\overline{V}-\overline{XB}) \cdot \overline{Hi}$	Inches^2
W	$\overline{WB} \cdot \overline{Hi}$	Dot product $\overline{WB} \cdot \overline{Hi}$	Inches^2

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE SPH

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
B	$\overline{DX}\cdot\overline{WB}$	B coefficient of the quadratic equation solution used to compute RIN and ROUT	Inches
C	$\overline{DX}^2-R^2$	C coefficient of the quadratic equation solution used to compute RIN and ROUT	Inches^2
DIS	B^2-C	Quantity under the radical of the quadratic equation solution used to compute RIN and ROUT	Inches^2
DX,DY,DZ	---	x, y and z coordinates of the vector $\overline{DX}$	Inches
I2	---	Pointer to the location of the radius in the ASTER array	ND
ITEMP	---	Location in ASTER array which contains the vertex coordinates	ND
R	R	Radius of sphere	Inches

LIST OF SYMBOLS AND ABBREVIATIONS (SIMULATION MODEL)

SUBROUTINE RCC

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
AMBD	λ'	Value of λ' from the equation $\tau S^2 - 2S\lambda' + \mu' - 0$	Inches
AMBDA	λ	Coefficient of 2S from the quadratic equation of the RCC	Inches
CM	---	RIN intersection with a planar surface	Inches
CP	---	ROUT intersection with a planar surface	Inches
DEN	τ	Coefficient of S^2 from the quadratic equation	ND
DISC	$\lambda^2 - \mu$	Quantity under the radical of the quadratic equation used to compute RIN and ROUT	Inches^2
F1	---	A quantity used to determine if RIN or ROUT lies within the boundary of the RCC	Inches
H(3)	$\overline{H}$	x, y, and z coordinates of the height vector $\overline{H}$	Inches
HH	$\overline{H} \cdot \overline{H}$	Dot product of the height vector $\overline{H}$ of the RCC	Inches^2
I	---	Index that represents an x, y, or z coordinate	ND
IH	---	x, y, and z coordinates of the light vector $\overline{H}$	Inches
IRR	---	Location index for the radius R of the RCC in the ASTER array	ND
IV	---	x, y, and z coordinates of the vertex $\overline{V}$	Inches
LCM	---	Temporary storage location for the surface number of RIN for a planar surface	ND

LIST OF SYMBOLS AND ABBREVIATIONS
SUBROUTINE RCC (Concluded) (SIMULATION MODEL)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
LCP	---	Temporary storage location for the surface number of ROUT for a planar surface	ND
POT	$(\overline{XB}-\overline{V})^2$	Portion of the expression for solving for μ	Inches^2
R	R	Radius of the right circular cylinder	Inches
R1	$\lambda - \sqrt{\lambda^2-\mu}$	Temporary location in the subroutine of RIN for the quadratic surface	Inches
R2	$\lambda + \sqrt{\lambda^2-\mu}$	Temporary location in the subroutine of ROUT for the quadratic surface	Inches
RSQ	R^2	Value of radius squared	Inches^2
SD	$\sqrt{\lambda^2 - \mu}$	Value $\sqrt{\lambda^2-\mu}$ from the expression $S = \lambda \pm \sqrt{\lambda^2-\mu}$	Inches
TOP	$\overline{WB} \cdot (\overline{XB}-\overline{V})$	Portion of the expression for solving for λ	Inches
UM	μ'	Value of μ' from the quadratic equation $\tau S^2-2S\lambda'+\mu' = 0$	Inches
UMU	μ	$\mu = \mu'/\tau$	Inches
V(3)	V	x, y, and z coordinates of the vertex $\overline{V}$	Inches
VPH	$\overline{H} \cdot (\overline{V}-\overline{XB})$	Quantity used in solving for λ and μ	Inches^2
WH	$\overline{WB} \cdot \overline{H}$	Quantity used in solving for γ and λ and RIN and ROUT on the planar surfaces	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE REC

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A(3)	$\overline{A}$	x, y, and z coordinates of the semi-major axis $\overline{A}$	Inches
AA	$\overline{A}\cdot\overline{A}$	Dot product of the semi-major axis $\overline{A}$	$Inches^2$
AAAA	$(\overline{A}\cdot\overline{A})^2$	Square of the dot product $\overline{A}\cdot\overline{A}$	$Inches^4$
AMBD	λ'	Coefficient of 2S from equation $\tau S^2+2\lambda' S+\mu'=0$	ND
AMBDA	λ'/τ	Coefficient, λ, of 2s from equation $S^2+2\lambda S+\mu=0$	ND
B(3)	$\overline{B}$	x, y, and z coordinates of the semi-minor axis $\overline{B}$	Inches
BB	$\overline{B}\cdot\overline{B}$	Dot product of the semi-minor axis $\overline{B}$	$Inches^2$
BBBB	$(\overline{B}\cdot\overline{B})^2$	Square of the dot product $\overline{B}\cdot\overline{B}$	$Inches^4$
CM	---	RIN intersection with a planar surface	Inches
CP	---	ROUT intersection with a planar surface	Inches
DEN	τ	Coefficient of S^2 from the quadratic equation of the REC	ND
DISC	$\lambda^2-\mu$	Quantity under the radical of the quadratic equation used in solving for RIN and ROUT on the quadratic surface	$Inches^2$
F1	---	Quantity used to determine if RIN or ROUT lies within the boundary of the REC	Inches
H(3)	$\overline{H}$	x, y, and z coordinates of the height vector $\overline{H}$	Inches

LIST OF SYMBOLS AND ABBREVIATIONS

SUBROUTINE REC (Continued) (SIMULATION MODEL)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
HH	$\overline{H}\cdot\overline{H}$	Dot product, $\overline{H}\cdot\overline{H}$, of the height vector $\overline{H}$	Inches^2
IA	---	Pointer to the coordinates of the semi-major axis $\overline{A}$	ND
IB	---	Pointer to the coordinates of the semi-minor axis $\overline{B}$	ND
IH	---	Pointer to the coordinates of the height vector $\overline{H}$	ND
IV	---	Pointer to the coordinates of the vertex $\overline{V}$	ND
LCM	---	Temporary storage location for the surface number of RIN for a planar surface	ND
LCP	---	Temporary storage location for the surface number of ROUT for a planar surface	ND
LOC	---	Index used to retrieve pointers to the location of the coordinates of the REC	ND
R1	$\lambda-\sqrt{\lambda^2-\mu}$	Temporary storage location of RIN for the quadratic surface	Inches
R2	$\lambda+\sqrt{\lambda^2-\mu}$	Temporary storage location of ROUT for the quadratic surface	Inches
SD	$\sqrt{\lambda^2-\mu}$	Value $\sqrt{\lambda^2-\mu}$ from the expression $S = \lambda\pm\sqrt{\lambda^2-\mu}$	Inches
UM	μ'	Value of μ' from the quadratic equation $\tau S^2-2S\lambda' + \mu' = 0$	ND
UMU	μ	$\mu = \mu'/\tau$	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE REC (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
V(3)	$\overline{V}$	x, y, and z coordinates of the vertex $\overline{V}$	Inches
V1XB1	$\overline{V}_x-\overline{XB}_x$	x component of the vector $\overline{V}-\overline{XB}$	Inches
V2XB2	$\overline{V}_y-\overline{XB}_y$	y component of the vector $\overline{V}-\overline{XB}$	Inches
V3XB3	V_z-XB_z	z component of the vector $\overline{V}-\overline{XB}$	Inches
VPA	$\overline{A}\cdot(\overline{V}-\overline{XB})$	Dot product of the semi-major axis $\overline{A}$ and the vector from $\overline{XB}$ to the vertex of the REC $(\overline{V}-\overline{XB})$	Inches^2
VPB	$\overline{B}\cdot(\overline{V}-\overline{XB})$	Dot product of the semi-minor axis $\overline{B}$ and the vector from $\overline{XB}$ to the vertex of the REC $(\overline{V}-\overline{XB})$	Inches^2
VPH	$\overline{H}\cdot(\overline{V}-\overline{XB})$	Dot product of the height vector $\overline{H}$ and the vector from $\overline{XB}$ to the vertex of the REC $(\overline{V}-\overline{XB})$	Inches^2
VPHHH	$\overline{H}\cdot(\overline{V}-\overline{XB})+\overline{H}\cdot\overline{H}$	Sum of the dot product $\overline{H}\cdot(\overline{V}-\overline{XB})$ and the dot product of the height vector $\overline{H}\cdot\overline{H}$	Inches
WBA	$\overline{WB}\cdot\overline{A}$	Dot product of the semi-major axis $\overline{A}$ and the direction cosines of the ray $\overline{WB}$	Inches
WBAWBA	$(\overline{WB}\cdot\overline{A})^2$	Square of the dot product $\overline{WB}\cdot\overline{A}$	Inches^2
WBB	$\overline{WB}\cdot\overline{B}$	Dot product of the semi-minor axis $\overline{B}$ and the direction cosines of the ray $\overline{WB}$	Inches
WBBWBB	$(\overline{WB}\cdot\overline{B})^2$	Square of the dot product $\overline{WB}\cdot\overline{B}$	Inches^2
WH	$\overline{WB}\cdot\overline{H}$	Quantity used for determining the direction of the ray with respect to the planar surfaces, and it is used in solving for RIN and ROUT of the planar surfaces	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TRC

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
AMBD	λ'	Coefficient, λ', of 2S from equation $\tau S2-2\lambda' S+\mu'=0$	ND
AMBDA	λ'/τ	Coefficient, , of 2S from equation $S^2-2\lambda S+\mu=0$	Inches
CM	--	Temporary storage location of RIN for a planar surface	Inches
CP	--	Temporary storage location of ROUT for a planar surface	Inches
DEN	τ	Coefficient, τ, of S^2 from equation $\tau S^2-2\lambda' S+\mu'=0$	ND
DISC	λ^2-u	Quantity under the radical of the quadratic equation used in solving for RIN and ROUT on the quadratic surface	Inches^2
F1	--	Variable used to determine if an intersect lies within the boundaries of the TRC	Inches
H(3)	$\overline{H}$	x, y, and z coordinates of height vector $\overline{H}$	Inches
HH	$\overline{H}\cdot\overline{H}$	Dot product of height vector $(\overline{H}\cdot\overline{H})$	Inches^2
IH	--	Pointer to the coordinates of the height vector	ND
INTR1	--	Counter for counting the number of intersects for RIN	ND
INTR2	--	Counter for counting the number of intersects for ROUT	ND
INTSEC	--	Counter for counting the number of valid intersects	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TRC (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IRB	--	Pointer to the value of the radius of the base	ND
IRTOP	--	Pointer to the value of the radius of the top	ND
IV	--	Pointer to the coordinates of the vertex	ND
LOC	--	Location of the pointers to the radii of the top and bottom of the TRC	ND
PVPV	$(\overline{V}-\overline{XB})^2$	Dot product of vector $(\overline{V}-\overline{XB})$ $(\overline{V}-\overline{XB})$	Inches^2
R1	$\lambda-\sqrt{\lambda^2-\mu}$	Temporary storage location of RIN or ROUT for the quadratic surface	Inches
R2	$\mu'/2\lambda'$ or $\lambda+\sqrt{\lambda^2-\mu}$	Temporary storage location of RIN or ROUT for the quadratic surface	Inches
RB	R_B	Radius at vertex $\overline{V}$	Inches
RBRTVP	C_2	Portion of the constant term used in solving the quadratic equation	Inches
RT	R_T	Radius at $\overline{V}+\overline{H}$	Inches
RTRB	R_T-R_B	Difference between the radius of the upper base radius and the radius of the lower base	Inches
SD	$\sqrt{\lambda^2-\mu}$	Value $\sqrt{\lambda^2\ \mu}$ from the expression $S = \lambda\pm\sqrt{\lambda^2-\mu}$	Inches
UM	μ'	Value of the constant term from the quadratic equation $\tau S^2-2\lambda' S+\mu'=0$	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TRC (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
UMU	μ	$\mu=\mu'/\tau$ from the equation $S^2-2\lambda S+\mu=0$	ND
V(3)	$\overline{V}$	x, y, and z coordinate of the vertex $\overline{V}$	Inches
V1XB1	$\overline{V}_x-\overline{XB}_x$	x component of vector $(\overline{V}-\overline{XB})$	Inches
V2XB2	$\overline{V}_y-\overline{XB}_y$	y component of vector $(\overline{V}-\overline{XB})$	Inches
V3XB3	$\overline{V}_z-\overline{XB}_z$	z component of vector $(\overline{V}-\overline{XB})$	Inches
VPH	$(\overline{V}-\overline{XB})\cdot\overline{H}$	Dot product of vector $(\overline{V}-\overline{XB})$ and height vector $\overline{H}$	Inches^2
VPHHH	$(\overline{V}-\overline{XB})\cdot\overline{H}$ $+\overline{H}\cdot\overline{H}$	Sum of dot product of vector $(\overline{V}-\overline{XB})$ and height vector $\overline{H}$ plus dot product of height vector $\overline{H}$	Inches^2
VPW	$(\overline{V}-\overline{XB})\cdot\overline{WB}$	Dot product of vector $(\overline{V}-\overline{XB})$ and direction cosines of the ray	Inches
WH	$\overline{WB}\cdot\overline{H}$	Dot product of direction cosines of the ray and height vector $\overline{H}$	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ELL

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A1	$2\,\overline{D1} \cdot \overline{WB}$	Vector dot product $2\,\overline{D_1} \cdot \overline{WB}$	Inches
A2	$2\,\overline{D2} \cdot \overline{WB}$	Vector dot product $2\,\overline{D_2} \cdot \overline{WB}$	Inches
AA	B	Second term of Equation (64)	ND
ALAM1	λ	Coefficient of quadratic equation used to compute RIN and ROUT	ND
ALAMD	B^2-1	Intermediate variable	ND
B1	$(D_1)^2$	Vector dot product $\overline{D_1}^2$	$Inches^2$
B2	$(D_2)^2$	Vector dot product $\overline{D_2}^2$	$Inches^2$
BB	A	First term of Equation (64)	ND
C	C	Length of ellipsoid major axis	Inches
D1X, D1Y, D1Z	D1	x, y, and z coordinates of $\overline{XB}$ - ($\overline{\text{Foci}}$ A)	Inches
D2X, D2Y, D2Z	D2	x, y, and z coordinates of $\overline{XB}$ - ($\overline{\text{Foci}}$ B)	Inches
DISCRM	$\lambda^2-\mu$	Quantity under radical of quadratic equation solution	$Inches^2$
FOCIA(3)	$\overline{F}_a$	x, y, and z coordinates of one of the ellipsoid's foci	Inches
FOCIB(3)	$\overline{F}_b$	x, y, and z coordinates of one of the ellipsoid's foci	Inches
IRR	---	Location where the ellipsoid's major axis length is stored	ND
IV1	---	Location of the coordinates of one of the ellipsoid's foci	ND

LIST OF SYMBOLS AND ABBREVIATIONS

SUBROUTINE ELL (Concluded) (SIMULATION MODEL)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IV2	---	Location of the coordinates of the other ellipsoid's foci	ND
SQRTDI	$\sqrt{\lambda^2-\mu}$	Square root of $(\lambda^2-\mu)$ for solving for RIN and ROUT	Inches
U	μ	Coefficient of quadratic equation used to compute RIN and ROUT	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE RAW

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
AG	A2G1 + A1G2	Denominator for computing S_2	Inches^3
ASQ(1)	Ai	Dot product $\overline{H_1} \cdot \overline{H_1}$	Inches^2
ASQ(2)	Ai	Dot product $\overline{H_2} \cdot \overline{H_2}$	Inches^2
ASQ(3)	Ai	Dot product $\overline{H_3} \cdot \overline{H_3}$	Inches^2
CM	---	Intermediate storage location for RIN	Inches
CP	---	Intermediate storage location for ROUT	Inches
G(1)	Gi	Dot product $\overline{WB} \cdot \overline{H_1}$	Inches
G(2)	Gi	Dot product $\overline{WB} \cdot \overline{H_2}$	Inches
G(3)	Gi	Dot product $\overline{WB} \cdot \overline{H_3}$	Inches
H1(3)	$\overline{H_1}$	x, y and z coordinates of vector $\overline{H_1}$	Inches
H2(3)	$\overline{H_2}$	x, y and z coordinates of vector $\overline{H_2}$	Inches
H3(3)	$\overline{H_3}$	x, y and z coordinates of the vector $\overline{H_3}$	Inches
I	---	Index used to determine if intersections with sides 1 and 3 are possible	ND
IH1	---	Pointer to the coordinates of the $\overline{H_1}$ vector	ND
IH2	---	Pointer to the coordinates of the $\overline{H_2}$ vector	ND
IH3	---	Pointer to the coordinates of the $\overline{H_3}$ vector	ND

LIST OF SYMBOLS AND ABBREVIATIONS

SUBROUTINE RAW (Concluded) (SIMULATION MODEL)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IV	---	Pointer to the coordinates of the vertex	ND
K	---	Variable used to determine if variables CP and CM have been updated with a computed intersection distance	ND
L	---	Variable used to determine if variables CP and CM have been updated with the computed intersection distance	ND
LOC	---	Index used to retrieve pointers to the location of the coordinates of the RAW	ND
PV(1)	Pi	Dot product $(\overline{XB} - \overline{V}) \cdot \overline{H_1}$	$Inches^2$
PV(2)	Pi	Dot product $(\overline{XB} - \overline{V}) \cdot \overline{H_2}$	$Inches^2$
PV(3)	Pi	Dot product $(\overline{XB} - \overline{V}) \cdot \overline{H_3}$	$Inches^2$
PV(4)	-(P1A2+P2A1)	Part of the numerator for computing S2	ND
TEMP	---	Intermediate variable used as temporary storage	Inches
TOP	A1A2-P1A2-P2A1	Numerator for computing S2	ND
V(3)	$\overline{V}$	x, y and z coordinates of the vector $\overline{V}$	Inches
XB1V1, XB2V2, XB3V3	$\overline{XB}-\overline{V}$	x, y and z coordinates of the vector difference $\overline{XB} - \overline{V}$	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ARB

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
AA (I,1) AA (I,2) AA (I,3) AA (I,4)	Ai Bi Ci Di	Coefficients of the equation for the ith face of the ARB	ND
D	Di	The D coefficient of the ith plane equation	ND
I	---	The ith face of the ARB	ND
J	---	The jth face of the ARB	ND
K	---	The kth coordinate (x, y, or z)	ND
L1	---	Intermediate variable used to store the face number containing the first computed intersection	ND
L2	---	Intermediate variable used to store the face number containing the second computed intersection	ND
LC	---	Pointer to the coefficients of the ith plane	ND
LD	---	Pointer to the constant term of the ith plane	ND
LOC	---	Index used to retrieve pointers to the ARB constants	ND
S	Si	Distance to the intersect for the ith face of the ARB	Inches
S1	---	Distance for the first intersect computed	Inches
S2	---	Distance for the second intersect computed	Inches
SDEN	---	Denominator of the equation for computing the distance to the ith face of the ARB	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ARB (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
SNUM	---	Numerator of the equation for computing the distance to the ith face of the ARB	ND
T	---	Intermediate variable used to test the equation for computing the distance to the ith face of the ARB	ND
T1	---	Intermediate variable used to test the equation for computing the distance to the ith face of the ARB	ND
XP	$\overline{XI}i$	x, y, and z coordinates of the point of intersection on the ith plane	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TEC

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A	$\gamma R3+R1(1-\gamma)$	Semi-major axis of the intersection ellipse	Inches
AA(3)	$\overline{AA}$	x, y, and z components of the semi-major axis unit vector of the base ellipse	Inches
ALPHA	$\frac{\overline{H}\cdot\overline{N}}{\overline{WB}\cdot\overline{N}}$	Distance along the ray from the plane of the base ellipse to the plane of the top ellipse	Inches
AMBDA	λ	Coefficient of 2S in the equation $\tau S^2+2\lambda S+\mu=0$	ND
ASQ	$(\gamma R3+R1(1-\gamma))^2$	Square of the semi-major axis of the intersection ellipse	$Inches^2$
B	$\gamma R4+R2(1-\gamma)$	Semi-minor axis of the intersection ellipse	Inches
BB(3)	$\overline{BB}$	x, y, and z coordinates of the semi-minor axis unit vector of the base ellipse	Inches
BETA	$\frac{(\overline{V}-\overline{XB})\cdot\overline{N}}{\overline{WB}\cdot\overline{N}}$	Distance along the ray from start point $\overline{XB}$ to plane of base ellipse	Inches
BSQ	$(\gamma R4+R2(1-\gamma))^2$	Square of the semi-minor axis of the intersection ellipse	$Inches^2$
DEN	τ	Coefficient of S^2 in the equation $\tau S^2+2\lambda S+\mu=0$	ND
DISC	$\lambda^2-\tau\mu$	Value under the radical of the quadratic equation used in solving for RIN and ROUT on the quadratic surface	ND
F	---	Intermediate variable used for determining if an intersect either lies between the two planes, or lies within the ellipse of a plane	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TEC (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
F1	---	Distance from the center of the top or base ellipse to the intersection along the $\bar{A}$ axis	Inches
F2	---	Distance from the center of the top or base ellipse to the intersection along the $\bar{B}$ axis	Inches
GAMMA	$-\frac{(\bar{V}-\overline{XB})\cdot\bar{N}}{\bar{H}\cdot\bar{N}}$	Ratio on normal of height of hit	ND
H(3)	$\bar{H}$	x, y, and z components of the height vector H	Inches
HDA	$\bar{H}\cdot\bar{A}$	Dot product of the height vector $\bar{H}$ and the semi-major axis unit vector $\bar{A}$ of the base ellipse	Inches
HDB	$\bar{H}\cdot\bar{B}$	Dot product of the height vector $\bar{H}$ and the semi-minor axis unit vector $\bar{B}$ of the base ellipse	Inches
HDN	$\bar{H}\cdot\bar{N}$	Dot product of the height vector $\bar{H}$ and the normal unit vector $\bar{N}$	Inches
HN(3)	$\bar{N}$	x, y, and z components of the normal unit vector $\bar{N}$	Inches
I	---	Variable used to count the intersects with the plane surfaces	ND
IA	---	Pointer to the coordinates of the semi-major axis unit vector	ND
IH	---	Pointer to the coordinates of the height vector	ND
IN	---	Pointer to the coordinates of the normal unit vector	ND
IR1	---	Pointer to the coordinates of the semi-major axis radius of the base ellipse	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TEC (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IR2	---	Pointer to the coordinates of the semi-minor axis radius of the base ellipse	ND
IR3	---	Pointer to the value of the ratio of the base ellipse to the top ellipse	ND
IV	---	Pointer to the coordinates of the vertex	ND
LI	---	Temporary storage location for surface number of RIN for a planar surface	ND
LO	---	Temporary storage location for surface number of ROUT for a planar surface	ND
LOC	---	Pointer to the location of TEC data in the ASTER array	ND
R1	R1	Length of the semi-major axis of the base ellipse	Inches
R2	R2	Length of the semi-minor axis of the base ellipse	Inches
R2SQ	$(R2)^2$	Square of the length of the semi-minor axis of the base ellipse	$Inches^2$
R3	R3	Length of the semi-major axis of the top ellipse	Inches
R4	R4	Length of the semi-minor axis of the top ellipse	Inches
RR	RR	Ratio of the larger to the smaller ellipse	ND
S	---	Temporary storage location for RIN and ROUT when verifying intersections	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TEC (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
S1	---	Temporary storage location for the distance to the entry intersect for a quadratic surface	Inches
S2	---	Temporary storage location for the distance to the exit intersect for a quadratic surface	Inches
SI	---	Temporary storage location for the distance to the entry intersect for a plane surface	Inches
SO	---	Temporary storage location for the distance to the exit intersect for a plane surface	Inches
T	---	Temporary storage location used to interchange the values of two variables that represent the two intersections with the quadratic surface	Inches
T1	---	Value of the quadratic equation for the smaller root	Inches
T2	---	Value of the quadratic equation for the larger root	Inches
TA	$(\overline{V}-\overline{XB})\cdot\overline{A}$ $+\gamma(\overline{H}\cdot\overline{A})$	Quantity used in solving the coefficient of 2S, λ, and in solving the constant term, μ, in the equation $\tau S^2+2\lambda S+\mu=0$	Inches
TA1	$\alpha\overline{WB}\cdot\overline{A}-\overline{H}\cdot\overline{A}$	Quantity used in solving for the coefficients of S^2 and 2S in the equation $\tau S^2+2\lambda S+\mu=0$	Inches
TA2	$(\overline{V}-\overline{XB})\cdot\overline{A}-\beta\overline{WB}\cdot\overline{A}$	Quantity used in solving for the constant term μ and the coefficient $\tau S^2+2\lambda S+\mu=0$	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TEC (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
TAU	$(R1/R2)^2$	Square of the ratio of the length of the semi-major axis of the base ellipse to the length of the semi-minor axis of the base ellipse	ND
TB	$(\overline{V}-\overline{XB})-\overline{B}+\gamma(\overline{H}\cdot\overline{B})$	Quantity used in solving the coefficient of 2S, λ, and in solving the constant term, μ, in the equation $\tau S^2+2\lambda S+\mu=0$	Inches
TB1	$\alpha\overline{WB}\cdot\overline{B}-\overline{H}\cdot\overline{B}$	Quantity used in solving for and λ in the equation $\tau S^2+2\lambda S+\mu=0$	Inches
TB2	$(\overline{V}-\overline{XB})\cdot\overline{B}-\overline{WB}\cdot\overline{B}$	Quantity used in solving for λ and μ in the equation $\tau S^2+2\lambda S+\mu=0$	Inches
TR2SQ	$(R1/R2)^2(R2)^2$	Quantity used in solving for λ and μ in the equation $\tau S^2+2\lambda S+\mu=0$	$Inches^2$
TR4R2	$(R1/R2)^2$ $(R2-R4)^2$	Quantity used in solving for τ in the equation $\tau S^2+2\lambda S+\mu=0$	$Inches^2$
TRR4R2	$(R1/R2)^2(R2)^2-(R1/R2)^2(R2)(R4)$	Quantity used in solving for λ in the equation $\tau S^2+2\lambda S+\mu=0$	$Inches^2$
UM	μ	Value of the constant term in the equation $\tau S^2+2\lambda S+\mu=0$	ND
VXB(3)	$\overline{V}-\overline{XB}$	x, y, and z components of the vector $\overline{V}-\overline{XB}$	Inches
VXBDA	$(\overline{V}-\overline{XB})\cdot\overline{A}$	Dot product of the vector $\overline{V}-\overline{XB}$ and the semi-major axis unit vector $\overline{A}$ of the base ellipse	Inches
VXBDB	$(\overline{V}-\overline{XB})\cdot\overline{B}$	Dot product of the vector $\overline{V}-\overline{XB}$ and the semi-minor axis unit vector $\overline{B}$ of the base ellipse	Inches
VXBDN	$(\overline{V}-\overline{XB})\cdot\overline{N}$	Dot product of the vector $\overline{V}-\overline{XB}$ and the normal unit vector $\overline{A}$	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TEC (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
WDA	$\overline{WB} \cdot \overline{A}$	Dot product of the direction cosines of the ray unit vector $\overline{WB}$ and the semi-major axis unit vector $\overline{A}$ of the base ellipse	ND
WBD	$\overline{WB} \cdot \overline{B}$	Dot product of the direction cosines of the ray unit vector $\overline{WB}$ and the semi-minor axis unit vector $\overline{B}$ of the base ellipse	ND
WDN	$\overline{WB} \cdot \overline{N}$	Dot product of the direction cosines of the ray unit vector $\overline{WB}$ and the normal unit vector $\overline{N}$	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ARS

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
ALPHA	α	Value of α from matrix equation: $\alpha(\overline{U}-\overline{W})+\beta(\overline{V}-\overline{W})-S\cdot\overline{WB}=\overline{XB}-\overline{W}$	ND*
BETA	β	Value of β from matrix equation: $\alpha(\overline{U}-\overline{W})+\beta(\overline{V}-\overline{W})-S\cdot\overline{WB}=\overline{XB}-\overline{W}$	ND
D	---	Denominator of matrix equation: $\alpha(\overline{U}-\overline{W})+\beta(\overline{V}-\overline{W})-S\cdot\overline{WB}=\overline{XB}-\overline{W}$	ND
DALPHA	---	Numerator for solving for α in matrix equation: $\alpha(\overline{U}-\overline{W})+\beta(\overline{V}-\overline{W})-S\cdot\overline{WB}=\overline{XB}-\overline{W}$	ND
DBETA	---	Numerator for solving for β in matrix equation: $\alpha(\overline{U}-\overline{W})+\beta(\overline{V}-\overline{W})-S\cdot\overline{WB}=\overline{XB}-\overline{W}$	ND
DS	---	Numerator for solving for S in matrix equation: $\alpha(\overline{U}-\overline{W})+\beta(\overline{V}-\overline{W})-S\cdot\overline{WB}=\overline{XB}-\overline{W}$	ND
GAMMA	$1-\alpha-\beta$	Value of γ from equation: $\overline{XP}=\overline{XB}+S\cdot\overline{WB}=\alpha\cdot\overline{U}+\beta\cdot\overline{V}+\gamma\cdot\overline{W}$	ND
HIT(20)	---	Array for storing intersect distances (largest to smallest) while solving ray intersection with ARS	Inches
I	---	DO loop index for hit number	ND
ISURF(20)	---	Array for storing intersected triangle number of ARS. Positive or negative integer for exit or entry intersect, respectively	ND
IT	---	DO loop index for triangle number of ARS	ND
J	---	Number of intersects save area when intersect with larger distance has been found	ND
JSURF	---	Intersected triangle number	ND

*Non-dimensional

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ARS (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
L	---	DO loop index for clearing RIN/ROUT ARS section of ASTER array	ND
LOC	---	Pointer for data in ARS section of ASTER array	ND
LOCARS	---	Pointer to beginning location of ARS data in ASTER array	ND
LOCHTS	---	Pointer to location of hit table in ASTER array	
L1	---	Lower limit of DO loop for clearing hit table in ASTER array	ND
L2	---	Upper limot of DO loop for clearing hit table in ASTER array	ND
NHIT	---	Number of intersects of current ray with ARS	ND
NT	---	Total number of possible combinations of points that could form triangles on ARS surface	ND
ORMAL(3,20)	$\overline{WN}(3)$	Intermediate storage array for storing normal coordinates to intersected triangle	ND
S	S	Distance from current origin of ray to intersected triangle	Inches
UW(3)	$\overline{U}-\overline{W}$	Array for storing coordinates of vector $\overline{U}-\overline{W}$ for computed triangle	Inches
VW(3)	$\overline{V}-\overline{W}$	Array for storing coordinates of vector $\overline{V}-\overline{W}$ for computed triangle	Inches
W(3)	$\overline{W}$	Array for storing coordinates of vector $\overline{W}$ for computed triangle	Inches
WN(3)	---	Array for storing coordinates of normal vector to surface of computed triangle	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE ARS (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
WXB(3)	$\overline{W}-\overline{XB}$	Array for storing coordinates of vector from point $\overline{W}$ of computed triangle to current origin of ray, $\overline{XB}$	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TOR

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
COEF(4)	B,C,D,E	Four-element array for coefficients of quadratic equation	ND
I	---	Index for RT array	ND
IN	---	Location of coordinates of normal of torus in ASTER array	ND
IR1	---	Location of major radius of torus in ASTER array	ND
IR2	---	Location of minor radius of torus in ASTER array	ND
IV	---	Location of coordinates of center of torus in ASTER array	ND
LOC	---	Location of packed word with radii of torus	ND
NR	---	Number of real intersects with torus	ND
R1	r_1	Major radius of torus	Inches
R1SQ	r_1^2	Square of major radius of torus	Inches^2
R2	r_2	Minor radius of torus	Inches
R2SQ	r_2^2	Square of minor radius of torus	Inches^2
RSAVE	---	Scalar quantity for shifting $(\overline{XB}-\overline{V})$ along ray to insure correct solution of quartic equation	ND
RT(4)	---	Four-element array for roots of quartic equation	Inches
T	---	Temporary storage location for exchanging values in two other storage locations	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TOR (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
TERM	$(\overline{XB}-\overline{C})^2-r_1^2-r_2^2$	Intermediate value for solving coefficients of quartic equation	Inches^2
WDN	$\overline{WB}\cdot\overline{N}$	Dot product of normal with direction cosines of ray	Inches
XBC(3)	$(\overline{XB}-\overline{C})$	Coordinates of vector from center of torus to ray origin	Inches
XBCDN	$(\overline{XB}-\overline{C})\cdot\overline{N}$	Dot product of vector from center of torus to ray origin with normal to torus	Inches
XBCDW	$(\overline{XB}-\overline{C})\cdot\overline{WB}$	Dot product of vector from center of torus to ray origin with direction cosines of ray	Inches
XBCXBC	$(\overline{XB}-\overline{C})^2$	Dot product of vector from center of torus to ray origin with itself	Inches^2
XN(3)	$\overline{n}$	Coordinates of unit normal vector of torus	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE QRTIC

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A	$\sqrt{\frac{a^2}{4} - b + 2w}$	Square root of coefficient of x^2 of biquadratic equation	ND
ASQ	$\frac{a^2}{4} - b + 2w$	Coefficient of x^2 of biquadratic equation	ND
B	$\sqrt{w-d}$	Square root of constant term of biquadratic equation	ND
BSQ	B^2	Coefficient of constant term of biquadratic equation	ND
C(4)	a,b,c,d	Coefficients of quartic equation	ND
C1SQ	a^2	Square of coefficient of x^3 of quartic equation	ND
DISC	See Equations (203) and (204)	Discriminate of quadratic Equations (203) and (204)	Inches^2
I	---	DO loop index for number of roots from cubic equation	ND
N	---	Number of real roots	ND
NN	---	Number of real roots in cubic equation	ND
R(4)	$x_{1,2,3,4}$	Array for storing roots of quartic equation	ND
REAL	$\frac{-\frac{a}{2} \pm e}{2}$	Real part of quadratic equation	Inches
ROOT	---	Real root of cubic equation	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE QRTIC (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
RR(3)	---	Three-element array with roots of cubic equation	ND
SQROOT	$\sqrt{DISC}$	Radical part of quadratic equation	Inches
T	$\frac{a^2}{4} - b$	Part of ASQ	ND
TWOAB	aw - c	Coefficient of x of biquadratic equation	ND

LIST OF SYMBOLS AND ABBREVIATIONS (SIMULATION MODEL)

SUBROUTINE CUBIC

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A	$\sqrt[3]{-\frac{q}{2} + \sqrt{Q}}$	Expression for solving the cubic equation for one real root and two conjugate complex roots	ND
AB	A + B	Expression for solving the cubic equation for one real root and two conjugate complex roots	ND
ACU	$-\frac{q}{2} + \sqrt{Q}$	Quantity under the radical of A	ND
B	$\sqrt[3]{-\frac{q}{2} - \sqrt{Q}}$	Expression for solving the cubic equation for one real root and two conjugate complex roots	ND
BCU	$-\frac{q}{2} - \sqrt{Q}$	Quantity under the radical of B	ND
C	a, b, or c	Coefficients of x^2, x, and the constant term	ND
C1SQ	a^2	Square of the coefficient of x^2	ND
C3	$\frac{a}{3}$	Value of the constant term of the cubic equation	ND
DISC	$4p^3 + 27q^2$	Expression for solving for Q	ND
HALFQ	q/2	One-half the value of Q	ND
N	---	Integer variable that represents the number of real roots in the cubic equation	ND
P	$b - \frac{a^2}{3}$	Expression for solving for Q	ND
PHI3	$\phi/3$	One-third the angle	Radians
Q	$\left(\frac{p}{3}\right)^3 + \left(\frac{q}{2}\right)^2$	Expression used to solve for A and B	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE CUBIC (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
R(1)	X_1	Argument for passing real root cubic equation	ND
R(2)	X_2	Argument for passing real root, or real part of complex root	ND
R(3)	X_3	Argument for passing real root, or imaginary part of complex root	ND
SQROOT	$\sqrt{Q}$	Expression used to solve for A and B	ND
T	$\sqrt{-p/3}$	Expression for solving for three real roots with two equal roots	ND
TT	$2\sqrt{-p/3}$	Expression for solving for two equal roots when there are three real roots	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE UN2

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
I3	---	Packed word from the Lth word of the MASTER array	ND
J1	---	Integer data item in the 15 bits previous to the last 15 bits of the two-item packed word	ND
J2	---	Integer data item in the last 15 bits of the two-item packed word	ND
L	---	Pointer to the location in the MASTER array of the packed word	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE UN3

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
I2	---	Packed word containing only J1 and J2 after J3 has been shifted out	ND
I3	---	Packed word from the Lth word of the MASTER array	ND
J1	---	Integer data item in the six bits previous to the six bits of J2	ND
J2	---	Integer data item in the six bits previous to the last 15 bits of J3	ND
J3	---	Integer data item in the last 15 bits of the packed word	ND
L	---	Pointer to the location in the MASTER array of the packed word	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE OPENK

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
I2	---	Packed word containing only J1 and J2 after J3 has been shifted out	ND
I3	---	Packed word from the Lth location of the ITR array	ND
J1	---	12 bits of the left integer data item of the packed word	ND
J2	---	Middle 12 bits of the integer data item of the packed word	ND
J3	---	Integer data item in the last 12 bits of the packed word	ND
L	---	Pointer to the location in the ITR array of the packed word	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE RAN

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
RAN	---	Random number between zero and one returned by Function URAN31	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE URAN31

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A1	---	Temporary fixed point storage location for computing the random number	ND
I	---	Integer variable that represents the argument of the function	ND
J	---	Temporary integer storage location used for computing the random number	ND
URAN31	---	Value of the random number between zero and one	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE CROSS

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
ANSWER(3)	---	Three-element array for the coordinates of the resultant vector from the cross product of two other vectors	Inches
FIRST(3)	---	Three-element array for the coordinates of the first argument vector	Inches
SECOND(3)	---	Three-element array for the coordinates of the second argument vector	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE DOT

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
DOT	---	Resultant scalar quantity from the dot product of two vectors	$Inches^2$
FIRST(3)	---	Three-element array for the coordinates of the first argument vector	Inches
SECOND(3)	---	Three-element array for the coordinates of the second argument vector	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE UNIT

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
TEMP	---	Scalar length of the vector	Inches
V(3)	---	Coordinates of either the original vector or the resultant unit vector	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE XDIST

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
XDIST	---	Distance between the two given points	Inches
XSUM	---	Temporary storage area for summing the coordinate distances between the two points	$Inches^2$
XA(3)	---	Three-element array for the coordinates of the first point	Inches
XB(3)	---	Three-element array for the coordinates of the second point	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE DCOSP

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
DIS	---	Scalar distance between points $\overline{XA}$ and $\overline{XB}$	Inches
WA(3)	---	Three-element array for the direction cosines or unit vector coordinates of the vector from point $\overline{XA}$ to $\overline{XB}$	Inches
XA(3)	---	Three-element array for the coordinates of the first point	Inches
XB(3)	---	Three-element array for the coordinates of the second point	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE TROPIC

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
CSPHI	---	Cosine of a random angle ϕ	ND
CSTHT	---	Cosine of a random angle θ	ND
SNPHI	---	Sine of a random angle ϕ	ND
SNTHT	---	Sine of a random angle θ	ND
T	---	Sum of the squares of two random numbers that is less than or equal to 1.0	ND
WP(3)	---	Three-element array containing the direction cosines of a random angle	ND
X1	---	Random number between zero and one	ND
X2	---	Random number between zero and one	ND
X1S	---	Square of random number X1	ND
X2S	---	Square of random number X2	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE S

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
I	---	Number of the RPP	ND
L	---	Location of the pointer data for a given side of a given RPP	ND
LL	---	Location of the coordinate for a given side of a given RPP	ND
N	---	Surface number of the RPP	ND
S	---	Coordinate of the given side of the given RPP number	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE RPP2

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
I	---	Index for a DO loop to test each potential abutting RPP	ND
I1	---	Number of a potential abutting RPP unpacked from the left position of the two-item packed word in the abutting RPP section	ND
I2	---	Number of a potential abutting RPP unpacked from the left position of the two-item packed word in the abutting RPP section	ND
IRP	---	Number of the potential RPP, or the RPP number returned to the calling program	ND
J	---	Index to represent an x, y, or z coordinate in a DO loop	ND
LOC	---	Variable used to represent the location of abutting RPP's in the MASTER array	ND
LOCAT	---	Pointer to the location of the abutting RPP list in the MASTER array	ND
LS	---	Control variable used to prevent a specific intersect coordinate from being tested with its respective plane coordinate	ND
LSURF	---	Surface number of the RPP where the intersect occurs	ND
M	---	Control integer used to determine which of two packed abutting RPP's is to be tested	ND
NC	---	Total number of abutting RPP's	ND
XP	---	Coordinates of the intersect	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE (VOLUM)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
DOD	---	Horizontal dimension of cell on front plane of box	Inches
DSP	---	Vector from first ray origin to second ray origin in given column	Inches
DT	---	Vertical dimension of cell on front plane of box	Inches
IR1	---	Region number of pre-computed volume	ND
IRJ	---	Region number save area	ND
IRPRIM	---	Region number returned by Subroutine G1	ND
J	---	Index for column or plane number	ND
J1	---	Index for x, y, or z coordinate	ND
JIR	---	Region number save area	ND
J1	---	Number of vertical cells	ND
N2	---	Number of horizontal cells	ND
S1	---	Distance to next region returned by Subroutine G1	Inches
SUMV	---	Sum of the computed volumes in box	$Inches^3$
TESTDN	---	Vertical distance to next region	Inches
TESTOV	---	Horizontal distance to next region	Inches
VASTER	---	Array for accumulating ray distance through each region within box	Inches
VR	---	Pre-computed volume of given region	$Inches^3$
WAB	---	Direction cosines of vector from plane to back plane of box	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE (VOLUM) (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
WOB	---	Direction cosines of horizontal vector from vertex across front plane of box	ND
WTB	---	Direction cosines of vertical vector from vertex across front plane of box	ND
XA(3)	---	Coordinates of point on back plane	Inches
XO(3)	---	Coordinates of lower left corner of front plane of box	Inches
XP(3)	---	Coordinates of intersect with next region	Inches
XPERC	---	Percent error of computed volume and pre-computed volume of given region	ND
XT(3)	---	Coordinates of upper right corner of front plane of box	Inches
XTEMP(3)	---	Temporary storage for direction cosines	ND
XV(3)	---	Coordinates of vertex of box	Inches
XVDIS	---	Distance from front plane to back plane of box	Inches

LIST OF SYMBOLS AND ABBREVIATIONS (SIMULATION MODEL)

SUBROUTINE AREA

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
A	A	Azimuth angle of grid plane	Degrees
AR	α	Azimuth angle of grid plane	Radians
AREAC	---	Area of cell in grid plane	Inches^2, Feet^2, Centi-meters^2, or Meters^2
AREAUN	---	Measurement units of presented area	ND
BLANK	---	Two Hollerith blanks	ND
CA	$\cos \alpha$	Cosine of azimuth angle	ND
CE	$\cos \theta$	Cosine of elevation angle	ND
CELL2	D/2	Half the dimension of a cell side	Inches, Feet, Centi-meters, or Meters
CELLUN	---	Measurement units of cell	ND
CONVRT(1,1)	---	Conversion factor for converting square inches to square inches (= 1)	ND
CONVRT (1,2)	---	Conversion factor for converting square inches to square feet (= 0.0069444444444444)	ND
CONVRT(1,3)	---	Conversion factor for convering square inches to square centi-meters (= 6.451625806)	ND
CONVRT(1,4)	---	Conversion factor for converting square inches to square meters (= 0.0006451625806)	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE AREA (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
CONVRT(2,1)	---	Conversion factor for converting square feet to square inches (= 144)	ND
CONVRT(2,2)	---	Conversion factor for converting square feet to square feet (= 1)	ND
CONVRT(2,3)	---	Conversion factor for converting square feet to square centimeters (= 929.0341161)	ND
CONVRT(2,4)	---	Conversion factor for converting square feet to square meters (= 0.09290341161)	ND
CONVRT(3,1)	---	Conversion factor for converting square centimeters to square inches (= 0.15499969)	ND
CONVRT(3,2)	---	Conversion factor for converting square centimeters to square feet (= 0.001076386736)	ND
CONVRT(3,3)	---	Conversion factor for converting square centimeters to square centimeters (= 1)	ND
CONVRT(3,4)	---	Conversion factor for converting square centimeters to square meters (= 0.0001)	ND
CONVRT(4,1)	---	Conversion factor for converting square meters to square inches (= 1549.9969)	ND
CONVRT(4,2)	---	Conversion factor for converting square meters to square feet (= 10.7636736)	ND
CONVRT(4,3)	---	Conversion factor for converting square meters to square centimeters (= 10000)	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE AREA (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
CONVRT(4,4)	---	Conversion factor for converting square meters to square meters (= 1)	ND
E	E	Elevation angle of grid plane	Degrees
ENGTH	ENGTH	Back-off distances or origin from grid plane	Inches
ER	θ	Elevation angle of grid plane	Radians
GROUND	---	z coordinate of ground level	Inches
HHBB	---	Two Hollerith blanks for testing for blank fields on card	ND
HHCM	---	Hollerith code for centimeters (CM)	ND
HHFT	---	Hollerith code for feet (FT)	ND
HHIN	---	Hollerith code for inches (IN)	ND
HHMB	---	Hollerith code for meters (M)	ND
I	---	Index for representing component code	ND
ICODE	---	Component code of region material	ND
IDENT	---	Identification code of region material	ND
IH	---	Random number for computing random horizontal point in given cell	ND
II	---	Variable for representing row number of grid cell	ND
IRPRIM	---	Next region number returned by Subroutine G1	ND
IRSTRT	---	Starting region of grid rays	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE AREA (Continued)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IV	---	Random number for computing random vertical point in given cell	ND
J	---	Variable for representing column number of grid cell	ND
KK	---	Index for representing cell number of grid	ND
KL	---	Total number of cells in grid	ND
LAREA	---	Beginning location of presented areas indexed by component code	ND
LAREA1	---	Last location of presented areas indexed by component code	ND
LOC	---	Location of specific data in MASTER-ASTER array	ND
NHIT	---	Number of rays that hit target	ND
NSTART	---	Beginning grid cell number	ND
NX	N_x	Number of horizontal cells in grid plane	ND
NY	N_y	Number of vertical cells in grid plane	ND
RADIAN	R	One degree expressed in radians	Radians
SA	$\sin \alpha$	Sine of azimuth angle	ND
SE	$\sin \theta$	Sine of elevation angle	ND
SUMA	---	Total presented area of target	$Inches^2$, $Feet^2$, $Centimeters^2$, or $Meters^2$
TYPEUN(4)	---	Four-element array containing Hollerith codes IN, FT, CM, and M, respectively	ND

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

SUBROUTINE AREA (Concluded)

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
WB(3)	$\overline{WB}$	Direction cosines of ray	ND
WP(3)	---	Isotropic random direction cosines returned by Subroutine TROPIC	ND
XB(3)	$\overline{XB}$	Coordinates of origin of ray	Inches
XBS(3)	$\overline{X}_p$	Coordinates of original position of origin on grid cell before back-off	Inches
XSHIFT	XSHIFT	Distance target origin and grid plane center is effectively shifted in the X direction	Inches
YSHIFT	YSHIFT	Distance target origin and grid plane center is effectively shifted in the Y direction	Inches
ZSHIFT	ZSHIFT	Distance target origin and grid plane center is effectively shifted in the Z direction	Inches

LIST OF SYMBOLS AND ABBREVIATIONS
(SIMULATION MODEL)

TESTG

Symbol or Abbreviation	Equivalent in Math Model	Definition	Units
IRAY	---	Index for number of rays to be processed	ND
IRFIN	---	Region number of end point	ND
IRSTRT	---	Region number of starting point	ND
NRAYS	---	Total number of rays to be processed	ND
RANGE	---	Distance between two points	Inches
S1	---	Distance to next region	Inches
XBF	---	Coordinates of end point	Inches
XP	---	Coordinates of intersect with next region	Inches

SECTION IV

SOURCE LISTING

SIMULATION SOURCE DECK

This section contains a listing of the FORTRAN statements that make up the program deck (Figures 74 through 114).

SAMPLE PROBLEM DECK

Figure 115 shows a listing of the sample problem deck data. A description of the sample problem is contained in Volume I, User Manual.

```
      DIMENSION A(6)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/TEMPOR/XS(6),X(6),IX(8),IT(10),IA(9),IN(9)
      COMMON/WALT/LIRFO,NG1ERR
      COMMON/CONTRL/ITESTG,IRAYSK,IENTLV,IVOLUM,IWOT,ITAPE8,NO,IYES
      COMMON/ENGEOM/LEGEOM
      COMMON/SIZE/NDQ
      COMMON/ERR/IERR0
      COMMON/RANDM/IRANDM
      EQUIVALENCE (ASTER,MASTER)
C
  901 FORMAT(1H1,32HTHIS IS THE 11 APR 69 VERSION OF /
     1         1H ,32HTHE BRLESC MAGIC PROGRAM  ****** //)
  902 FORMAT(16H BEGIN EXECUTION)
  903 FORMAT(8I10)
  904 FORMAT(1H0,10X,42HTHE TAPE 4 USED FOR THIS RUN HAS THE TITLE /
     1  10A6/)
  905 FORMAT(1H0,10HENTER GENI)
  906 FORMAT(1H0,12HLEAVING GENI)
  907 FORMAT(1H0,35HTERMINATION ON GEOMETRY INPUT ERROR,5X,5HIERR=,I5)
  908 FORMAT(1H1,15HTESTG IS CALLED)
  909 FORMAT(1H0,13HLEAVING TESTG)
  910 FORMAT(1H1,24HREGION TYPE DATA FOLLOWS, 8X,6HLIRFO=,I10/
     1         1H ,6HREGION,6X,4HCODE,6X,4HTYPE,6X,11HDESCRIPTION/)
  911 FORMAT(3I10,10X,6A6)
  912 FORMAT(I6,I10,I9,7X,6A6)
  913 FORMAT(1H0,23HNO ROOM FOR IDENT TABLE,5X,7HLEGEOM=,I7,5X,
     1  6HLIRFO=,I7)
  914 FORMAT(1H0,32HWRITE TAPE 1 OPTION IS SPECIFIED)
  915 FORMAT(I5,10X,10A6)
  916 FORMAT(1H1,11HENTER VOLUM)
  917 FORMAT(1H0,13HLEAVING VOLUM)
  918 FORMAT(1H ,6H 999.9)
  919 FORMAT(1H1,11HEND OF CASE,I5)
  925 FORMAT(1H1,32HNUM OF ASPECT ANGLES FOR GRID IS,I5)
  927 FORMAT(10I5)
  928 FORMAT(1H1,32HNUM OF ASPECT ANGLES FOR AREA IS,I5)
  929 FORMAT(1H0,31HNUMBER OF G1 ERRORS ENCOUNTERED,I5)
  930 FORMAT(1H0,31HNUMBER OF  0  ITEMS ENCOUNTERED,I5)
  999 FORMAT(1H0,10HEND OF RUN)
C
      IRANDM=0
      WRITE (6,901)
      WRITE (6,902)
C
C1    INITIALIZE CONSTANTS
C
      I15=2**15
      I30=2**30
      PINF=1.0E50
      NO=0
      IYES=1
      IERR=0
```

FIG. 74. Source Listing, MAIN Routine

```
      LBASE=1
      KLOOP=0
      NDQ=10000
C
C2    ENTER AND INITIALIZE OPTION PARAMETERS
C
      READ (5,903)IRDTP4,IWRTP4,ITESTG,IRAYSK,ICARDI,IENTLV,IVOLUM
      IF(IRDTP4.NE.0)IRDTP4=IYES
      IF(IWRTP4.NE.0)IWRTP4=IYES
      IF(ITESTG.NE.0)ITESTG=IYES
      IF(IRAYSK.NE.0)IRAYSK=IYES
      IF(ICARDI.NE.0)ICARDI=IYES
      IF(IENTLV.NE.0)IENTLV=IYES
      IF(IVOLUM.NE.0)IVOLUM=IYES
C
C3    ENTER TARGET GEOMETRY FROM INPUT TAPE 4
C
      IF(IRDTP4.EQ.NO) GOTO 10
      READ (4) LBASE,LEGEOM,NDQ,(ASTER(L),L=1,NDQ),LBODY,LREGD,LRIN
     1  LROT,LIO,LIRFO,NRPP,NBODY,NRMAX,PINF,IT
      WRITE (6,904)(IT(I),I=1,10)
      GOTO 20

C
C4    CLEAR MASTER-ASTER ARRAY
C
   10 DO 11 I=LBASE,NDQ
      ASTER(I)=0.
   11 CONTINUE
C
C5    ENTER AND PROCESS TARGET GEOMETRY VIA SUBROUTINE GENI
C
      WRITE (6,905)
      CALL GENI
      WRITE (6,906)
      IF(IERR.LE.0)GOTO 12
      WRITE (6,907)IERR
      STOP

C
C6    WRITE OUT TARGET GEOMETRY TO OUTPUT TAPE 4
C
   12 IF(IWRTP4.EQ.NO)GOTO 20
      WRITE (4) LBASE,LEGEOM,NDQ,(ASTER(L),L=1,NDQ),LBODY,LREGD,LRIN,
     1  LROT,LIO,LIRFO,NRPP,NBODY,NRMAX,PINF,IT

C
C7    CALL SUBROUTINE TESTG
C
   20 IF(ITESTG.EQ.NO)GOTO 30
      WRITE (6,908)
      CALL TESTG
      WRITE (6,909)
      ITESTG=NO
C
C8    CALL SUBROUTINE VOLUM
C
   30 IF(IVOLUM.EQ.NO)GOTO 40
      WRITE (6,916)
      CALL VOLUM
```

FIG. 74. (Contd.)

```
      WRITE (6,917)
      IVOLUM=NO
C
C9    REGION IDENTIFICATION DATA        FORMAT - / ICODE / IDENT /
C          IRN   = REGION NUMBER
C          ICODE = ITEM CODE
C          IDENT = SPACE CODE AND SPECIAL IDENTIFICATION
C                    SPECIAL IDENTIFICATION = 10,20,30,40,50,60,70,80,90
C                      NO IDENT CODE=0  SKIRT=10  ARMOR=20  TARGET=30
C                    SPACE CODES   EXTERIOR VOLUME = 1
C                                  INTERIOR VOLUME = -1,1-9,...,91-99
C
   40 LIRFO=NDQ-NRMAX-10
      IF(LIRFO.GT.LEGEOM)GOTO 41
      WRITE(6,913)LEGEOM,LIRFO
      STOP
   41 WRITE (6,910)LIRFO

C
C10   ENTER AND STORE REGION ID DATA
C
   42 READ(5,911)IRN,ICODE,IDENT,(A(I),I=1,6)
      IF(IRN.LE.0)GOTO 50
      WRITE (6,912)IRN,ICODE,IDENT,(A(I),I=1,6)
      IDENT=IDENT+1
      K=LIRFO+IRN-1
      MASTER(K)=ICODE*I15+IDENT
      GOTO 42
C
C11        NOAA    = NUMBER OF ASPECT ANGLES FOR SUBROUTINE GRID
C          ITAPE8  = SUPPRESS PRINTER OPTION
C          IWOT    = WRITE OPTION FOR TAPE 1
C          NAREA   = NUMBER OF ASPECT ANGLES FOR SUBROUTINE AREA
C
   50 READ (5,927)NOAA,IWOT,ITAPE8,NAREA
      IF(IWOT.NE.0)IWOT=IYES
      IF(ITAPE8.EQ.0)GOTO 51
      ITAPE8=NO
      GOTO 52
   51 ITAPE8=IYES
   52 IF(IWOT.EQ.NO)GOTO 60
      REWIND 1
      WRITE (6,914)
      WRITE(1,915)NOAA,(IT(I),I=1,10)
C
   60 IF(NOAA.LE.0)GOTO 70
      WRITE (6,925)NOAA
C
C12   CALL SUBROUTINE GRID FOR EACH ASPECT ANGLE
C
      DO 61 I=1,NOAA
      IERR=0
      IERR0=0
      CALL GRID
      IF(IWOT.EQ.IYES)WRITE(1,918)
      WRITE (6,919)I
      WRITE (6,929)IERR
      WRITE (6,930)IERR0
   61 CONTINUE
C
```

FIG. 74. (Contd.)

```
   70 IF(NAREA.LE.0)GOTO 99
      WRITE (6,928)NAREA
C
C13   CALL SUBROUTINE AREA FOR EACH ASPECT ANGLE
C
      DO 71 I=1,NAREA
      IERR=0
      CALL AREA
      WRITE (6,919)I
   71 CONTINUE
C
   99 WRITE (6,999)
      STOP
      END
C
C
```

FIG. 74. (Concluded)

```
      SUBROUTINE GENI
      DIMENSION ITY(11),IAN(8),IAA(8),FX(20),NBOD(11),
     1  NO0(3),NO1(3),NO2(3),O4(3),TT(3),TT1(3),TT2(3)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/TEMPOR/XS(6),X(6),IX(8),IT(10),IA(9),IN(9)
      COMMON/CONTRL/ITESTG,IRAYSK,IENTLV,IVOLUM,IWOT,ITAPE8,NO,IYES
      COMMON/SIZE/NDQ
      COMMON/UNCLE/NN,IC(4)
      COMMON/RRPP/LRPPD,LABUT
      COMMON/ENGEOM/LEGEOM
      EQUIVALENCE(ASTER,MASTER)
C
  901 FORMAT(1H0,24HSTART READING SOLID DATA)
  902 FORMAT(10A6)
  903 FORMAT(1H0,10A6/)
  904 FORMAT(7I10)
  905 FORMAT(4X,34HNO. OF RECTANGULAR PARALLELEPIPEDS,I10/
     1        4X,34HNO. OF SOLIDS                     ,I10/
     2        4X,34HMAX NO. OF REGIONS                ,I10)
  906 FORMAT(1H0,45X,32HRECTANGULAR PARALLELEPIPED INPUT)
  911 FORMAT(1H0,50X,22HDESCRIPTION OF  SOLIDS)
  912 FORMAT(3A1,A3,A4,6F10.5)
  913 FORMAT(1H0,6HITYPE ,A3,27H DOES NOT MATCH WITH AN ITY)
  914 FORMAT(I9,1X,3A1,3X,A3,A4,3X,8I5)
  915 FORMAT(I8,1X,3A1,2X,A3,A4,4X,6F12.5)
  916 FORMAT(25X,6F12.5)
  917 FORMAT(1H0,38HNO MORE ROOM FOR SOLID DATA     LDATA=,I10,
     1  5X,5HLBOT=,I10,5X,4HNDQ=,I10)
  918 FORMAT(1H0,25HFINISH READING SOLID DATA)
  919 FORMAT(1H0, 5HLREGD,7H  LREGL,7H  LENLV,7H   LRIN,7H   LROT,
     1     7H    LIO,7H LEGEOM/I5,6I7)
  920 FORMAT(1H1,36X,23HREGION COMBINATION DATA)
  921 FORMAT(I5,1X,9(A2,I5))
  922 FORMAT(1H0,30HERROR IN DESCRIPTION OF REGION,I5,
     19H IN FIELD,I2,5X,24HBODY NUM.GT.NRPP + NBODY)
  923 FORMAT(10X,9(1H(,A2,I5,1H),1X))
  924 FORMAT(I8,2X,9(1H(,A2,I5,1H),1X))
  925 FORMAT(1H0,30HILLEGAL OPERATOR IN ABOVE CARD,5X,A2,
     1  9H IN FIELD,I2)
  926 FORMAT(1H0,29HERROR IN REGION INPUT    IR=,I5,14H OR N.GT.NRMAX)
  927 FORMAT(1H0,39HNO MORE ROOM FOR REGION DATA     LDATA=,I10,
     1  5X,4HNDQ=,I10)
  928 FORMAT(1H0,26HFINISH READING REGION DATA)
  929 FORMAT(14H ERROR, REGION,I10,18H IS PART OF REGION,I10)
  930 FORMAT(24H FINISH CHECKING REGION ,I5)
  931 FORMAT(1H0,34HNO MORE ROOM FOR ENTER LEAVE TABLE,5X,
     1  6HLDATA=,I10,5X,4HNDQ=I10,5X,4HPASS,I2,5X,3HIR=,I10)
  932 FORMAT(1H0,28HTOTAL ROOM FOR GEOMETRY DATA,5X,7HLEGEOM=,I6)
  933 FORMAT(1H0,5HENTER,18I6/(23X,15I6))
  934 FORMAT(1H ,5HLEAVE,18I6/(23X,15I6))
  935 FORMAT(1H1,50X,18HBEGIN ARRAY OUTPUT/)
  936 FORMAT(3(3I6,1X,E11.4,3H S ))
```

FIG. 75. Source Listing, Subroutine GENI

```
  937 FORMAT(/)
  938 FORMAT(1H0,34HFINISH A PASS OF ENTER LEAVE TABLE,I5)
  940 FORMAT(10X,6F10.5)
  941 FORMAT(1H0,37HTERMINATION ON BAD REGION DESCRIPTION)
  942 FORMAT(1H0,32HERROR IN DESCRIPTION OF BODY NUM,I6/
     1     7H VECTOR,3F12.5,24H IS NOT PERPENDICULAR TO /
     2     7H VECTOR,3F12.5/)
  943 FORMAT(1H0,27HERROR IN DESCRIPTION OF TOR,5X,8HR2.GT.R1/)
  944 FORMAT(1H0,27HERROR IN DESCRIPTION OF TRC,5X,7HR1 = R2/)
  945 FORMAT(1H0,5HLBASE,7H  LRPPD,
     1     7H  LABUT,7H  LBODY,7H   LBOD,7H  LDATA,7H   LBOT,7H  LSCAL,
     2     7H  LTRIP,7H    NDQ/I5,9I7)
  946 FORMAT(1H1,17HENTER-LEAVE TABLE)
  947 FORMAT(1H0,11(2X,A3)/11I5)
  948 FORMAT(1H0,27HERROR IN DESCRIPTION OF TEC,5X,
     1     41HHEIGHT VECTOR IS PARALLEL TO BASE ELLIPSE)
C
      DATA ITY(1),ITY(2),ITY(3),ITY(4),ITY(5),ITY(6),ITY(7),ITY(8)
     1     / 3HBOX, 3HSPH, 3HRCC, 3HREC, 3HTRC, 3HELL, 3HRAW, 3HARB /
      DATA ITY(9),ITY(10),ITY(11)
     1     / 3HTEC, 3HTOR , 3HARS /
      DATA IAA(1),IAA(2),IAA(3),IAA(4),IAA(5),IAA(6),IAA(7),IAA(8)
     1     / 2H   , 2HOR , 2H R ,2HR   , 2HRA , 2HAR , 2H A , 2HA   /
      DATA IAN(1),IAN(2),IAN(3),IAN(4),IAN(5),IAN(6),IAN(7),IAN(8)
     1     /     4 ,    1 ,    1 ,    1 ,    2 ,    2 ,    3 ,    3 /
      DATA IBL/1H /
C
      DO 10 I=1,11
   10 NBOD(I)=0
C
C2    ENTER AND PRINT OUT TITLE OF THE PROBLEM
C
      WRITE (6,901)
      READ(5,902)(IT(I),I=1,10)
      WRITE  (6,903)(IT(I),I=1,10)
C
C3    ENTER AND PRINT OUT THE PROGRAM CONTROL PARAMETERS
C
      READ(5,904)NRPP,NTRIP,NSCAL,NBODY,NRMAX,IPRIN,IRCHEK
      WRITE(6,905)NRPP,NBODY,NRMAX
C
C4    RPP
C
      WRITE (6,906)
      LAR=1
C
C5    RPP DATA INPUT
C
      IF(NRPP.LE.0)GOTO 20
      CALL RPPIN(LAR)
      IF(IERR.GT.0)RETURN
C
C7    LBODY STORAGE       RESERVE 3*(NRPP+NBODY) WORDS
C       /          BODY NUMBER          / LOC OF POINTER TO BODY DATA/
C       / REGION ENTER TABLE POINTER / REGION LEAVE TABLE POINTER /
C       / NUM REGIONS IN ENTER TABLE / NUM REGIONS IN LEAVE TABLE /
C
   20 LBOT=NDQ-2
```

FIG. 75. (Contd.)

```
      L=LAR
      LBODY=L+1
      LDATA=LBODY+3*(NBODY+NRPP)
      LBOD=LDATA
C
   50 WRITE (6,911)
C
C10    ENTER DATA FOR BODY
C
      DO 370 N=1,NBODY
      NN=N+NRPP
      LS1=0
      READ(5,912) IC(1),IC(2),IC(3),ITYPE,IC(4),(FX(K),K=1,6)
      DO 51 I=1,11
      IF(ITYPE.EQ.ITY(I))GOTO 52
   51 CONTINUE
      WRITE (6,913)ITYPE
      STOP
   52 ITYPE=I
      NBOD(I)=NBOD(I)+1
      K=LBODY+3*(NRPP+N-1)
      MASTER(K)=ITYPE*I15+LDATA
C
C             BOX SPH RCC REC TRC ELL RAW ARB TEC TOR ARS
  200 GOTO(201,220,202,201,203,202,201,230,204,203,240),ITYPE
  201 LE=12
      GOTO 210
  202 LE= 7
      GOTO 210
  203 LE= 8
      GOTO 210
  204 LE=13
  210 WRITE (6,915)NN,IC(1),IC(2),IC(3),ITY(ITYPE),IC(4),(FX(J),J=1,6)
      READ(5,940) (FX(J),J=7,LE)
      WRITE (6,916) (FX(J),J=7,LE)
C            BOX SPH RCC REC TRC ELL RAW ARB TEC TOR ARS
      GOTO(290,300,300,290,285,270,290,300,260,250,300),ITYPE
C          SPH
  220 WRITE (6,915)NN,IC(1),IC(2),IC(3),ITY(ITYPE),IC(4),(FX(J),J=1,4)
      GOTO 300
C
C14    ENTER BODY DATA FOR ARB
C
  230 WRITE (6,915)NN,IC(1),IC(2),IC(3),ITY(ITYPE),IC(4),(FX(J),J=1,6)
      CALL ALBERT(FX,LBOT,NDQ,LS1)
      GOTO 360
C
C15    ENTER BODY DATA FOR ARS
C
  240 CALL ARIN(LBOT,LDATA,MASTER,ASTER,IWH)
      GOTO 360
C
C16         TOR      CONVERT NORMAL VECTOR TO UNIT VECTOR
C
  250 TT(1)=FX(4)
      TT(2)=FX(5)
      TT(3)=FX(6)
      CALL UNIT(TT)
      FX(4)=TT(1)
      FX(5)=TT(2)
      FX(6)=TT(3)
      IF(FX(7).GE.FX(8))GOTO 280
      WRITE (6,943)
      IERR=IERR+1
      GOTO 280
```

FIG. 75. (Contd.)

```
C
C17       TEC     VERIFY SEMI-MAJOR AND SEMI-MINOR AXES PERPENDICULAR
C
  260 FX(15)=FX(13)
      LE=15
      TT1(1)=FX(7)
      TT1(2)=FX(8)
      TT1(3)=FX(9)
      TT2(1)=FX(10)
      TT2(2)=FX(11)
      TT2(3)=FX(12)
      IF(ABS(DOT(TT1,TT2)).LE.0.01) GOTO 265
      WRITE (6,942)NN,TT1,TT2
      IERR=IERR+1
C
C18   COMPUTE SEMI-MAJOR AXIS LENGTH AND CONVERT
C     SEMI-MAJOR AXIS TO UNIT VECTOR
C
  265 FX(13)=SQRT(DOT(TT1,TT1))
      CALL UNIT(TT1)
      FX(10)=TT1(1)
      FX(11)=TT1(2)
      FX(12)=TT1(3)
C
C19   COMPUTE SEMI-MINOR AXIS LENGTH AND CONVERT
C     SEMI-MINOR AXIS TO UNIT VECTOR
C
      FX(14)=SQRT(DOT(TT2,TT2))
      CALL CROSS(TT,TT1,TT2)
      CALL UNIT(TT)
      MON=FX(4)*TT(1)+FX(5)*TT(2)+FX(6)*TT(3)
      IF(MON)267,266,268
  266 WRITE(6,948)
      IERR=IERR+1
      GOTO 268
  267 TT(1)=-TT(1)
      TT(2)=-TT(2)
      TT(3)=-TT(3)
  268 FX(7)=TT(1)
      FX(8)=TT(2)
      FX(9)=TT(3)
      GOTO 280
C
C20   COMPUTE FOCI FOR ELL
C
  270 IF(IC(4).EQ.IBL)GOTO 300
      ASQ=FX(4)*FX(4)+FX(5)*FX(5)+FX(6)*FX(6)
      C=SQRT(ASQ-FX(7)*FX(7))
      A=SQRT(ASQ)
      FX(7)=A+A
C
C21   COMPUTE X,Y,Z COMPONENTS OF FOCI
C
      CX=C*FX(4)/A
      CY=C*FX(5)/A
      CZ=C*FX(6)/A
C
C22   COMPUTE X,Y,Z COORDINATES OF FOCI
C
      FX(4)=FX(1)+CX
      FX(5)=FX(2)+CY
      FX(6)=FX(3)+CZ
```

FIG. 75. (Contd.)

```
      FX(1)=FX(1)-CX
      FX(2)=FX(2)-CY
      FX(3)=FX(3)-CZ
C
C23   PRINT OUT NEW INPUT
C
  280 WRITE (6,915)NN,IC(1),IC(2),IC(3),ITY(ITYPE),IC(4),(FX(J),J=1,6)
      WRITE (6,916)(FX(J),J=7,LE)
      GOTO 300
C
C24        TRC    VERIFY LOWER AND UPPER RADII NOT EQUAL
C
  285 IF(FX(7).NE.FX(8))GOTO 300
      WRITE (6,944)
      IERR=IERR+1
      GOTO 300
C
C25   VERIFY THAT VECTORS ARE PERPENDICULAR IF BOX, RAW, OR REC
C
  290 IF(ABS(FX(4)*FX(7)+FX(5)*FX(8)+FX(6)*FX(9)).LE.0.01)GOTO 291
      WRITE (6,942)NN,(FX(J),J=4,9)
      IERR=IERR+1
  291 IF(ABS(FX(4)*FX(10)+FX(5)*FX(11)+FX(6)*FX(12)).LE.0.01)GOTO 292
      WRITE (6,942)NN,FX(4),FX(5),FX(6),FX(10),FX(11),FX(12)
      IERR=IERR+1
  292 IF(ABS(FX(7)*FX(10)+FX(8)*FX(11)+FX(9)*FX(12)).LE.0.01)GOTO 300
      WRITE (6,942)NN,(FX(J),J=7,12)
      IERR=IERR+1
C
C26   STORE BODY DATA AND BODY DATA POINTERS IN MASTER=ASTER ARRAY
C
C           BOX SPH RCC REC TRC ELL RAW ARB TEC TOR ARS
  300 GOTO(310,320,330,310,340,330,310,230,350,340,240),ITYPE
C
C27   POINTER FORMAT    BOX RAW  / V  / H1 /       REC  / V  / M  /
C                                / H2   H3 /            / H1 / H2 /
C
  310 CALL SEE3(IWH,ASTER,MASTER,FX(1),FX(2),FX(3),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA)=IWH*I15
      CALL SEE3(IWH,ASTER,MASTER,FX(4),FX(5),FX(6),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA)=MASTER(LDATA)+IWH
      CALL SEE3(IWH,ASTER,MASTER,FX(7),FX(8),FX(9),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA+1)=IWH*I15
      CALL SEE3(IWH,ASTER,MASTER,FX(10),FX(11),FX(12),
     1  LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA+1)=MASTER(LDATA+1)+IWH
      LDATA=LDATA+2
      GO TO 360
C
C28   POINTER FORMAT    SPH  / V  / R  /
C
  320 CALL SEE3(IWH,ASTER,MASTER,FX(1),FX(2),FX(3),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA)=IWH*I15
      LS1=1
      CALL SEE3(IWH,ASTER,MASTER,FX(4),FX(4),FX(4),LBOT,LDATA,NDQ,LS1)
      LS1=0
      MASTER(LDATA)=MASTER(LDATA)+IWH
      LDATA=LDATA+1
      GOTO 360
```

FIG. 75. (Contd.)

```
C
C29     POINTER FORMAT    RCC  / V   / H  /         ELL  / F1 / F2 /
C                              /     / R  /              /    / L  /
C
  330 CALL SEE3(IWH,ASTER,MASTER,FX(1),FX(2),FX(3),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA)=IWH*I15
      CALL SEE3(IWH,ASTER,MASTER,FX(4),FX(5),FX(6),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA)=MASTER(LDATA)+IWH
      LS1=1
      CALL SEE3(IWH,ASTER,MASTER,FX(7),FX(7),FX(7),LBOT,LDATA,NDQ,LS1)
      LS1=0
      MASTER(LDATA+1)=IWH
      LDATA=LDATA+2
      GO TO 360
C
C30     POINTER FORMAT    TRC  / V   / H  /         TOR  / V  / N  /
C                              / RB  / RT /              / R1 / R2 /
C
  340 CALL SEE3(IWH,ASTER,MASTER,FX(1),FX(2),FX(3),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA)=IWH*I15
      CALL SEE3(IWH,ASTER,MASTER,FX(4),FX(5),FX(6),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA)=MASTER(LDATA)+IWH
      LS1=1
      CALL SEE3(IWH,ASTER,MASTER,FX(7),FX(7),FX(7),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA+1)=IWH*I15
      CALL SEE3(IWH,ASTER,MASTER,FX(8),FX(8),FX(8),LBOT,LDATA,NDQ,LS1)
      LS1=0
      MASTER(LDATA+1)=MASTER(LDATA+1)+IWH
      LDATA=LDATA+2
      IF(ITYPE.EQ.10)LDATA=LDATA+3
      GO TO 360
C
C31     POINTER FORMAT    TEC  / V   / H  /
C                              / N   / A  /
C                              / R1  / R2 /
C                              /     / RR /
C
  350 CALL SEE3(IWH,ASTER,MASTER,FX(1),FX(2),FX(3),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA)=IWH*I15
      CALL SEE3(IWH,ASTER,MASTER,FX(4),FX(5),FX(6),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA)=MASTER(LDATA)+IWH
      CALL SEE3(IWH,ASTER,MASTER,FX(7),FX(8),FX(9),LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA+1)=IWH*I15
      CALL SEE3(IWH,ASTER,MASTER,FX(10),FX(11),FX(12),
     1  LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA+1)=MASTER(LDATA+1)+IWH
      LS1=1
      CALL SEE3(IWH,ASTER,MASTER,FX(13),FX(13),FX(13),
     1  LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA+2)=IWH*I15
      CALL SEE3(IWH,ASTER,MASTER,FX(14),FX(14),FX(14),
     1  LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA+2)=MASTER(LDATA+2)+IWH
      CALL SEE3(IWH,ASTER,MASTER,FX(15),FX(15),FX(15),
     1  LBOT,LDATA,NDQ,LS1)
      LS1=0
      MASTER(LDATA+3)=IWH
      LDATA=LDATA+4
```

FIG. 75. (Contd.)

```
C
C32    CHECK IF ANY MORE ROOM FOR SOLID DATA
C
  360 IF(LDATA.LT.NDQ)GOTO 370
      WRITE (6,917)LDATA,LBOT,NDQ
      STOP
  370 CONTINUE
      WRITE (6,918)
      WRITE(6,947)ITY,NBOD
      WRITE (6,945)LBASE,LRPPD,LABUT,LBODY,LBOD,LDATA,LBOT,LSCAL,LTRIP,N
     ADQ
C
C33    TRANSFER ASTER(LBOT TO NDQ) TO ASTER(LDATA TO LDATA+LSUB)
C
      LD=LDATA-1
      LSUB=LBOT-LD-1
      DO 375 I=LBOT,NDQ
      ASTER(LDATA)=ASTER(I)
      LDATA=LDATA+1
  375 CONTINUE
C
C34    UNPACK POINTER WORDS AND RECOMPUTE POINTERS TO
C      COMPENSATE FOR TRANSFER
C
      K=LBODY+3*(NRPP+NBODY)
      DO 390 I=K,LD
      CALL UN2(I,I1,I2)
      IF(I1.NE.0)I1=I1-LSUB
      IF(I2.NE.0)I2=I2-LSUB
      MASTER(I)=I1*I15+I2
  390 CONTINUE
C
C35    REGION STORAGE
C
      WRITE (6,920)
      N=0
      J=0
      LREGD=LDATA
      LDATA=LDATA+NRMAX
      LREGL=LDATA
C
C36    ENTER REGION DATA
C
  400 READ(5,921)IR,(IA(I),IN(I),I=1,9)
C
C37    CHECK VALIDITY OF REGION DATA
C
      DO 410 I=1,9
      IF(IABS(IN(I)).LE.NBODY+NRPP)GOTO 410
      WRITE (6,922)IR,I
      J=J+1
  410 CONTINUE
C
C38    PACK AND STORE REGION POINTER DATA
C         LREGD     / POINTER TO REGION / NUMBER OF BODIES IN REGION /
C
      IF(IR)440,420,421
  420 WRITE (6,923)(IA(I),IN(I),I=1,9)
      GOTO 430
  421 N=N+1
```

FIG. 75. (Contd.)

```
      WRITE (6,924)IR,(IA(I),IN(I),I=1,9)
      M=LREGD+N-1
      MASTER(M)=LDATA*I15
C
C39   CHECK AND CONVERT OPERATOR TO NUMERICAL VALUE
C
  430 DO 435 I=1,9
      DO 431 K=1,8
      IF(IA(I).EQ.IAA(K))GOTO 432
  431 CONTINUE
      WRITE (6,925)IA(I),I
      STOP
  432 IA(I)=IAN(K)
      IF(IN(I))433,435,434
  433 IA(I)=4+IA(I)
      IN(I)=-IN(I)
C
C40   PACK AND STORE REGION DATA      / OPERATOR / BODY NUMBER /
C
  434 MASTER(LDATA)=IA(I)*I15+IN(I)
      LDATA=LDATA+1
      MASTER(M)=MASTER(M)+1
      IF(LDATA.LT.NDQ)GOTO 435
      WRITE (6,927)LDATA,NDQ
      STOP
  435 CONTINUE
      GOTO 400
C
C41   END REGION READ - TEST NUMBER OF REGIONS
C
  440 IF(N.GE.NRMAX)GOTO 441
      WRITE (6,926)IR
      STOP
  441 IF(J.LE.0)GOTO 442
      WRITE (6,941)
      STOP
  442 WRITE (6,928)
C
C42   TEST FOR REGION CHECK OPTION, CHECK REGION DATA IF NOT ZERO
C      (ERROR IF ANY POINT CAN BE IN MORE THAN ONE REGION)
C
      IF(IRCHEK.EQ.NO)GOTO 500
      WRITE (6,937)
      LL=0
      MIS=0
C
      DO 456 I=1,NRMAX
      JJ=I+1
      DO 455 J=JJ,NRMAX
      KRI=LREGD+I-1
      CALL UN2(KRI,LOCI,NUMI)
      KRJ=LREGD+J-1
      CALL UN2(KRJ,LOCJ,NUMJ)
      IF(NUMI.GE.NUMJ)GOTO 450
      IO=NUMI
      II=NUMI
      GOTO 451
  450 IO=NUMJ
      II=NUMI
      L=LOCI
```

FIG. 75. (Contd.)

```
      LOCI=LOCJ
      LOCJ=L
C
  451 DO 453 KO=1,IO
      KLK=LOCI+KO-1
      CALL UN2(KLK,IOPO,NBO)
      DO 452 KI=1,II
      KLK=LOCJ+KI-1
      CALL UN2(KLK,IOPI,NBI)
      IF(IOPO.NE.IOPI)GOTO 452
      IF(NBO .NE.NBI )GOTO 452
      MIS=MIS+1
      GOTO 453
  452 CONTINUE
  453 CONTINUE
      IF(MIS.NE.II)GOTO 454
      WRITE (6,929) J,I
      LL=LL+1
  454 MIS=0
  455 CONTINUE
      WRITE (6,930) I
  456 CONTINUE
      IF(LL.GT.0)STOP
      WRITE (6,937)
C
C43   PREPARE REGION LEAVE TABLE (IS=-1) AND REGION ENTER TABLE (IS=+1)
C
  500 IS=-1
      NN=NBODY+NRPP
      LENLV=LDATA
      DO 590 MMM=1,2
      DO 580 I=1,NN
      M=LBODY+3*(I-1)
      IF(IS.GE.0)GO TO 510
      MASTER(M+1)=MASTER(M+1)+LDATA
      GO TO 520
  510 MASTER(M+1)=MASTER(M+1)+LDATA*I15
C
  520 DO 570 J=1,NRMAX
      ITEMP=LREGD+J-1
      CALL UN2(ITEMP,LOC,NC)
      CALL UN2(LOC,IOP,DUM)
      DO 560 N=1,NC
      MM=LOC+N-1
      CALL UN2(MM,IOPER,NUM)
      IF(NUM.NE.I)GOTO 560
      IF(IOP.EQ.1.OR.IOP.EQ.5)GOTO 540
      IF(IOPER.GT.4)GOTO 530
      IF(IS-1)560,550,560
  530 IF(IS+1)560,551,560
  540 IF(IS.LT.0)GOTO 551
  550 MASTER(M+2)=MASTER(M+2)+I15
      GO TO 552
  551 MASTER(M+2)=MASTER(M+2)+1
  552 MASTER(LDATA)=J
      LDATA=LDATA+1
      IF(LDATA.LT.NDQ)GOTO 570
      WRITE (6,931)LDATA,NDQ,MMM,I
      STOP
  560 CONTINUE
```

FIG. 75. (Contd.)

```
  570 CONTINUE
  580 CONTINUE
      WRITE (6,938)MMM
      IS=IS+2
  590 CONTINUE
C
C44   RESERVE SPACE FOR RIN STORAGE, ROUT STORAGE, AND
C       SUBROUTINE G1 TEMPORARY STORAGE
C
      L1=LDATA-1
      NN=NRPP+NBODY
      LRIN=LDATA+1
      LROT=LRIN+NN
      LIO=LROT+NN
      LEGEOM=LIO+NN
      WRITE (6,932)LEGEOM
      WRITE (6,919)LREGD,LREGL,LENLV,LRIN,LROT,LIO,LEGEOM
C
C45   TEST REGION ENTER/LEAVE TABLE PRINT OPTION
C
      IF(IENTLV.EQ.NO)RETURN
      WRITE(6,946)
      NBNR=NBODY+NRPP
C
C46   PRINT OUT REGION ENTER/LEAVE TABLES
C
      DO 600 N=1,NBNR
      LOC=LBODY+3*(N-1)
      LOC=LOC+1
      CALL UN2(LOC,LENT,LEAV)
      LOC=LOC+1
      CALL UN2(LOC,NENT,NEAV)
      J1=LENT
      J2=LENT+NENT-1
      WRITE (6,933)N,J1,J2,(MASTER(K),K=J1,J2)
      J1=LEAV
      J2=LEAV+NEAV-1
      WRITE (6,934)N,J1,J2,(MASTER(K),K=J1,J2)
  600 CONTINUE
C
C47   TEST MASTER-ASTER ARRAY PRINT OPTION
C
      IF(IPRIN.EQ.0)RETURN
      WRITE (6,935)
C
C48   PRINT OUT MASTER-ASTER ARRAY TO END OF REGION ENTER/LEAVE TABLES
C
      DO 620 K=LBASE,L1,3
      IK=K
      IK2=K+2
      M=0
      DO 610 I=IK,IK2
      M=M+1
      CALL UN2(I,I1,I2)
      NO1(M)=I1
      NO2(M)=I2
      O4(M)=ASTER(I)
      NO0(M)=I
  610 CONTINUE
      WRITE (6,936)(NO0(L),NO1(L),NO2(L),O4(L),L=1,3)
```

FIG. 75. (Contd.)

```
  620 CONTINUE
      RETURN
      END
C
C
```

FIG. 75. (Concluded)

```
      SUBROUTINE RPPIN(LAR)
      DIMENSION X(6)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1    LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/RRPP/LRPPD,LABUT
      EQUIVALENCE(MASTER,ASTER)
C
  910 FORMAT(6E12.5)
  920 FORMAT(I8,17X,6F12.5)
  930 FORMAT(1H0,27HERROR IN DESCRIPTION OF RPP,I5,5X,10HMIN.GE.MAX)
  940 FORMAT(1H0,27HERROR IN DESCRIPTION OF RPP,7X,I10,10X,I10)
  950 FORMAT(10X,7HSURFACE,I5,8X,2E20.6)
C
      IERR=0
      N=1
C
C1    LBASE - BEGINNING LOCATION OF RPP POINTERS   RESERVE 12 WORDS/RPP
C           / I / J /    I (POINTER TO LIST OF ABUTTING RPP-S)
C           /   / K /    J (NUMBER OF RPP-S THAT ABUT THIS SURFACE)
C                        K (POINTER TO BOUNDARY COORDINATE FOR SURFACE)
C
      I=LBASE+12*NRPP
C
C2    LRPPD - BEGINNING LOCATION OF RPP BOUNDARY COORDINATES
C             THAT ARE POINTED TO BY K  (LBASE + 12 * NRPP)
C
      LRPPD=I
C
C3    ENTER BOUNDARY COORDINATES OF RPP
C
   10 READ(5,910)(X(J),J=1,6)
      WRITE (6,920)N,(X(J),J=1,6)
C
C4    VERIFY MINIMUM BOUNDARY COORDINATE LESS THAN CORRESPONDING
C     MAXIMUM BOUNDARY COORDINATE
C
      DO 20 J=1,6,2
      IF(X(J).LT.X(J+1))GOTO 20
      WRITE (6,930)N
      STOP
   20 CONTINUE
C
C5    STORE BOUNDARY COORDINATES BEGINNING AT
C     LOCATION LBASE + (12 * NRPP)
C
      DO 33 J=1,6
      II=LBASE+12*NRPP
      L=LBASE+12*(N-1)+2*(J-1)
   30 IF(II.LT.I)GOTO 31
      ASTER(I)=X(J)
      MASTER(L+1)=I
      I=I+1
      GOTO 33
C
C6    CHECK FOR AND ELIMINATE REDUNDANT BOUNDARY COORDINATES
C
   31 IF(X(J).EQ.ASTER(II))GOTO 32
      II=II+1
      GOTO 30
```

FIG. 76. Source Listing, Subroutine RPPIN

```
   32 MASTER(L+1)=II
   33 CONTINUE
      IF(N.GE.NRPP)GOTO 40
      N=N+1
      GOTO 10
C
C7    LABUT - BEGINNING LOCATION OF LIST OF ABUT RPP-S PACKED 2/WORD
C             I POINTS HERE
C             J CONTAINS NUMBER IN LIST
C
   40 LABUT=I
      LAST=I-1
      L=LAST
C
C8    SEARCH FOR ABUTTING RPP-S TO SURFACE J OF RPP I
C
      DO 57 I=1,NRPP
      DO 57 N=1,6
      LL=0
      M=1
      K=LBASE+12*(I-1)+2*(N-1)
      MASTER(K)=(L+1)*I15+MASTER(K)
      NC=3*N-1-4*(N/2)
C
C9    DETERMINE IF RPP J HAS ABUTTING SURFACE TO RPP I
C
      DO 56 J=1,NRPP
      IF(I.EQ.J)GOTO 56
      IF(S(I,N).NE.S(J,NC))GOTO 56
C
C10   COMPARE BOUNDARY COORDINATES OF RPP-S I ANU J
C
      DO 53 K=1,3
      NN=N+NC
      K41=4*K-1
      IF(NN.EQ.K41)GOTO 53
      K2=2*K
      K21=K2-1
      IF(S(I,K21).GT.S(J,K21))GOTO 50
      IF(S(J,K21).LT.S(I,K2 ))GOTO 53
   50 IF(S(I,K21).GE.S(J,K2 ))GOTO 51
      IF(S(J,K2 ).LE.S(I,K2 ))GOTO 53
   51 IF(S(I,K2 ).GT.S(J,K2 ))GOTO 56
      IF(S(I,K21).LT.S(J,K21))GOTO 56
   53 CONTINUE
C
C11   STORE RPP NUMBER IN ABUTTING RPP LIST AND INCREMENT NUMBER
C
      M=-M
      IF(M.LT.0)GOTO 54
      MASTER(L)=MASTER(L)+J
      GOTO 55
   54 L=L+1
      MASTER(L)=J*I15
   55 LL=LL+1
   56 CONTINUE
      K=LBASE+12*(I-1)+2*(N-1)
      MASTER(K)=MASTER(K)+LL
   57 CONTINUE
C
```

FIG. 76. (Contd.)

```
      IF(NRPP.LE.1)GOTO 63
C
C12   TEST VALIDITY OF RPP DATA
C
C
      DO 62 J=1,6
      NRPP1=NRPP-1
      DO 61 I=1,NRPP1
      JJ=LBASE+12*(I-1)+2*(J-1)
      CALL UN2(JJ,IDUM,I2)
      I3=MASTER(JJ+1)
      IF(I2.NE.0)GOTO 61
      II=I+1
      DO 60 K=II,NRPP
      KK=LBASE+12*(K-1)+2*(J-1)
      CALL UN2(KK,IDUM,I5)
      I6=MASTER(KK+1)
      IF(I5.NE.0)GOTO 60
      IF(I3.EQ.I6)GOTO 60
      IERR=IERR+1
      WRITE (6,940)I,K
      WRITE (6,950)J,ASTER(I3),ASTER(I6)
   60 CONTINUE
      GOTO 62
   61 CONTINUE
   62 CONTINUE
   63 LAR=L
      RETURN
      END
C
C
```

FIG. 76. (Concluded)

```
      SUBROUTINE ALBERT(FX,LBOT,NDQ,LS1)
      DIMENSION IA(6,4),AA(8,3),F(4),FX(6)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGO,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      EQUIVALENCE(ASTER,MASTER)
C
  901 FORMAT(25X,6F12.5)
  902 FORMAT(10X,6(1X,4I1))
  903 FORMAT(10X,6E10.3)
  904 FORMAT(25X,6(4X,4I2))
  905 FORMAT(1H0,15HUNDEFINED PLANE)
  906 FORMAT(I5,10(E11.4))
  907 FORMAT(1H0,26HFOUR POINTS NOT IN A PLANE)
  908 FORMAT(1H0,25HERROR IN SIDE DESCRIPTION)
  909 FORMAT(1H0,16HDEGENERATE PLANE,I5)
C
C1    STORE COORDINATES OF FIRST TWO VERTICES IN ARRAY AA
C
      K=1
      DO 10 I=1,2
      DO 10 J=1,3
      AA(I,J)=FX(K)
      K=K+1
   10 CONTINUE
C
C2    ENTER COORDINATES OF REMAINING SIX VERTICES INTO ARRAY AA
C
      READ(5,903)((AA(I,J),J=1,3),I=3,8)
C
C3    ENTER ORDINAL NUMBERS OF PLANE VERTICES
C
      READ(5,902)((IA(I,J),J=1,4),I=1,6)
      WRITE  (6,901)((AA(I,J),J=1,3),I=3,8)
      WRITE  (6,904)((IA(I,J),J=1,4),I=1,6)
C
      DO 70 I=1,6
C
C4    RETRIEVE FIRST THREE VERTEX COORDINATES OF PLANE
C
      IX=IA(I,1)
      IY=IA(I,2)
      IZ=IA(I,3)
      X1=AA(IX,1)
      Y1=AA(IX,2)
      Z1=AA(IX,3)
      X2=AA(IY,1)
      Y2=AA(IY,2)
      Z2=AA(IY,3)
      X3=AA(IZ,1)
      Y3=AA(IZ,2)
      Z3=AA(IZ,3)
C
C5    COMPUTE COEFFICIENTS OF PLANE EQUATION
C
      D=X1*(Y2*Z3-Z2*Y3)-X2*(Y1*Z3-Z1*Y3)+X3*(Y1*Z2-Z1*Y2)
      A=(-Y2*Z3+Z2*Y3+Y1*Z3-Z1*Y3-Y1*Z2+Z1*Y2)
      B=(X2*Z3-Z2*X3-X1*Z3+X3*Z1+X1*Z2-Z1*X2)
      C=(Y2*X3-X2*Y3-Y1*X3+X1*Y3+Y1*X2-X1*Y2)
```

FIG. 77. Source Listing, Subroutine ALBERT

```
      D12=(X1-X3)**2+(Y1-Y3)**2+(Z1-Z3)**2
C
C6    TEST FOR DEGENERATE PLANE
C
      A2B2C2=A*A+B*B+C*C
      IF(A2B2C2.NE.0.)GOTO 21
      WRITE (6,909)I
      D=ABS(D)
      GOTO 61
C
C7    TEST FOR UNDEFINED PLANE
C
   21 D1210=D12*1.0E-12
      IF(A2B2C2.GT.D1210)GOTO 22
      WRITE (6,905)
      WRITE (6,906)I,A,B,C,D,D12
      IERR=IERR+1
      GOTO 70
   22 S=SQRT(A2B2C2)
      WX=A/S
      WY=B/S
      WZ=C/S
C
C8    RETRIEVE COORDINATES OF FOURTH VERTEX ON PLANE
C
      IC=IA(I,4)
      X4=AA(IC,1)
      Y4=AA(IC,2)
      Z4=AA(IC,3)
C
C9    COMPUTE DISTANCE TO PLANE OF FOURTH VERTEX
C
      D2=(-D-(A*X4)-(B*Y4)-(C*Z4))/((A*WX)+(B*WY)+(C*WZ))
      D22=D2*D2
C
C10   DETERMINE IF FOURTH VERTEX LIES ON PLANE OF FIRST THREE VERTICES
C
      IF(D22.LE.1.01)GOTO 30
      WRITE (6,907)
      IERR=IERR+1
      WRITE (6,906)I,A,B,C,D,D12,D2
      GOTO 70
C
   30 DO 31 K=1,4
      F(K)=0.
   31 CONTINUE
C
C11   COMPUTE VALUES OF OTHER FOUR VERTICES
C     WITH RESPECT TO PRESENT SIDE
C
      L=1
      DO 32 J=1,8
      IF(J.EQ.IX.OR.J.EQ.IY.OR.J.EQ.IZ.OR.J.EQ.IC)GOTO 32
      F(L)=A*AA(J,1)+B*AA(J,2)+C*AA(J,3)+D
      L=L+1
   32 CONTINUE
C
C12   COMPUTE NUMBER OF OTHER VERTICES ON EITHER
C     SIDE OF PLANE OR ON PLANE
C
```

FIG. 77. (Contd.)

```
      M=0
      N=0
      J=0
      DO 44 L=1,4
      IF(ABS(F(L)).LE.1.0E-6)GOTO 42
      IF(F(L))41,42,43
   41 M=M+1
      GOTO 44
   42 N=N+1
      GOTO 44
   43 J=J+1
   44 CONTINUE
C
C13   DETERMINE SIDE OF PLANE OTHER VERTICES ARE LOCATED
C
      IF(N.EQ.0)GOTO 51
      IF(M+N.EQ.4)GOTO 60
      IF(J+N.EQ.4)GOTO 61
      GOTO 52
   51 IF(M.EQ.4)GOTO 60
      IF(J.EQ.4)GOTO 61
   52 WRITE (6,908)
      WRITE (6,906)I,A,B,C,D,D12,D2,(F(L),L=1,4)
      IERR=IERR+1
      GOTO 70
C
   60 A=-A
      B=-B
      C=-C
      D=-D
C
C14   STORE PLANE COEFFICIENTS AND POINTERS
C
   61 CALL SEE3(IWH,ASTER,MASTER,A,B,C,LBOT,LDATA,NDQ,LS1)
      MASTER(LDATA)=IWH
      LS1=1
      CALL SEE3(IWH,ASTER,MASTER,D,D,D,LBOT,LDATA,NDQ,LS1)
      LS1=0
      MASTER(LDATA)=MASTER(LDATA)+IWH*I15
      LDATA=LDATA+1
   70 CONTINUE
      RETURN
      END
C
C
```

FIG. 77. (Concluded)

```
      SUBROUTINE ARIN(LBOT,LDATA)
C
C         SUBROUTINE READS, CHECKS, PROCESSES, AND STORES INPUT DATA
C         FOR THE ARS (ARBITRARY SURFACE)
C
      DIMENSION W(3),UW(3),VW(3),WN(3)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/UNCLE/NN,IC(4)
      EQUIVALENCE (MASTER,ASTER)
C
  901 FORMAT(1H ,I8,1X,3A1,2X,3HARS,2X,A4,2X,8X,
     1               37HNUMBER OF CURVES                          M=,I10 /
     2         1H ,33X,37HNUMBER OF POINTS PER CURVE              N=,I10 /
     3         1H ,33X,37HNUMBER OF POINTS IN                    MN=,I10 /
     4         1H ,33X,37HNUMBER OF POINTS STORED        NP=2N(M-1)=,I10 /
     5         1H ,33X,37HTOTAL STORAGE                 NSTR=4NP+82=,I10 )
  903 FORMAT(25X,6F12.4)
  904 FORMAT(10X,6F10.5)
  905 FORMAT(1H ,33X,34HNUMBER OF TRIANGLES DESCRIBED        ,I10 )
  906 FORMAT(1H ,33X,34HNUMBER OF NON-DEGENERATE TRIANGLES,I10 )
  910 FORMAT(1H0,43HERROR IN DESCRIPTION OF ARS    SOLID NUMBER,I5)
  911 FORMAT(5X,21HNUMBER OF POINTS IS 0)
  920 FORMAT(10X,2I10)
C
C         ENTER NUMBER OF CURVES AND NUMBER OF POINTS PER CURVE AND
C         COMPUTE NUMBER OF POINT TO BE STORED AND STORAGE REQUIREMENTS
C
      READ(5,920)M,N
      MN=M*N
      NP=2*N*(M-1)
      NSTR=4*NP+82
      WRITE(6,901)NN,IC(1),IC(2),IC(3),IC(4),M,N,MN,NP,NSTR
C
C         CHECK IF NUMBER OF POINTS IS 0
C
      IF(NP.GT.0)GOTO 10
      WRITE(6,910)NN
      WRITE(6,911)
      RETURN
C
C         RESERVE STORAGE IN MASTER-ASTER ARRAY FOR ARS DATA
C
   10 LBOT=LBOT-NSTR
      MASTER(LDATA)=LBOT
      LDATA=LDATA+1
      LOC=LBOT+82
C
C         ENTER AND STORE COORDINATE DATA OF ARS
C
      LOCC=LOC+4
      DO 230 I=1,M
      IF(I.EQ.M)LOC=LOCC
      L1=LOC
      L2=LOC+8*(N-1)
      READ(5,904)(ASTER(L),ASTER(L+1),ASTER(L+2),L=L1,L2,8)
      WRITE(6,903)(ASTER(L),ASTER(L+1),ASTER(L+2),L=L1,L2,8)
      IF(I.NE.M)WRITE(6,903)
      IF(I.EQ.1.OR.I.EQ.M)GOTO 220
```

FIG. 78. Source Listing, Subroutine ARIN

```
      DO 210 L=L1,L2,8
      ASTER(LOCC)=ASTER(L)
      ASTER(LOCC+1)=ASTER(L+1)
      ASTER(LOCC+2)=ASTER(L+2)
      LOCC=LOCC+8
  210 CONTINUE
  220 LOC=L2+8
  230 CONTINUE
C
C          STORE NUMBER OF POINTS STORED FOR ARS AND INITIALIZE LOCATION
C          FOR STORING NUMBER OF HITS FOR SHOTLINE
C
      MASTER(LBOT)=NP
      MASTER(LBOT+1)=0
C
C          ELIMINATE DEGENERATE TRIANGLES FOR GIVEN ARS DATA
C
      NT=NP-2
      WRITE(6,905)NT
      L1=LBOT+82
      L2=L1+4*(NT-1)
      DO 350 L=L1,L2,4
      W(1)=ASTER(L)
      W(2)=ASTER(L+1)
      W(3)=ASTER(L+2)
      UW(1)=ASTER(L+4)-W(1)
      UW(2)=ASTER(L+5)-W(2)
      UW(3)=ASTER(L+6)-W(3)
      VW(1)=ASTER(L+8)-W(1)
      VW(2)=ASTER(L+9)-W(2)
      VW(3)=ASTER(L+10)-W(3)
      CALL CROSS(WN,UW,VW)
      IF(DOT(WN,WN).GT.0.0001)GOTO 350
      NT=NT-1
      ASTER(L+3)=-1.0
  350 CONTINUE
C
      WRITE(6,906)NT
      WRITE(6,903)
      RETURN
      END
```

FIG. 78 (Concluded)

```
      SUBROUTINE SEE3(IWH,ASTER,MASTER,FX,FXX,FXXX,LBOT,LDATA,NDQ,LS1)
      DIMENSION ASTER(10000),MASTER(10000)
C
C1    TEST TO DETERMINE IF TRIPLET OR SCALAR DATA
C
      IF(LS1.NE.0)GOTO 50
C
C2    EXECUTE IF TRIPLET DATA
C
      IF(LBOT.GT.NDQ)GOTO 20
      NDQ2=NDQ-2
C
C3    SEARCH FOR EQUAL TRIPLET IN THE ASTER ARRAY
C
      DO 10 I=LBOT,NDQ2
      IF(ASTER(I).NE.FX)GOTO 10
      IF(ASTER(I+1).NE.FXX)GOTO 10
      IF(ASTER(I+2).NE.FXXX)GOTO 10
      IWH=I
      RETURN
   10 CONTINUE
C
C4    STORE TRIPLET PASSED BY ARGUMENT LIST
C
   20 ASTER(LBOT-1)=FXXX
      ASTER(LBOT-2)=FXX
      ASTER(LBOT-3)=FX
      LBOT=LBOT-3
      IWH=LBOT
      IF(LBOT.LE.LDATA)WRITE (6,30)LBOT,LDATA
      RETURN
C
   30 FORMAT(1H0,22HMEMORY OVERLAP IN SEE3,5X,5HLBOT=,I10,
     1  5X,6HLDATA=,I10)
C
C5    EXECUTE IF SCALAR QUANTITY
C     SEARCH FOR EQUAL SCALAR QUANTITY IN THE ASTER ARRAY
C
   50 DO 60 I=LBOT,NDQ
      IF(ASTER(I).NE.FX)GOTO 60
      IWH=I
      RETURN
   60 CONTINUE
C
C6    STORE SCALAR QUANTITY PASSED BY ARGUMENT LIST
C
      ASTER(LBOT-1)=FX
      LBOT=LBOT-1
      IWH=LBOT
      RETURN
      END
C
C
```

FIG. 79. Source Listing, Subroutine SEE3

```
      SUBROUTINE GRID
      DIMENSION WP(3)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/GTRACK/D1,D2,KHIT,LMAX,TR(200),XBS(3),IRSTRT,IENC,
     1  ITR(200),CA,CE,SA,SE
      COMMON/CAL/NIR,SLOS,ANGLE,NTYPE,SSPACE,L,XS(3),WS(3),
     1  TRAVEL,SN,V,H,IVIH
      COMMON/WALT/LIRFO,NG1ERR
      COMMON/HOYT/VREF,HREF
      COMMON/CELL/CELSIZ
      COMMON/CONTRL/ITESTG,IRAYSK,IENTLV,IVOLUM,IWOT,ITAPE8,NO,IYES
C
  901 FORMAT(8I10)
  902 FORMAT(6E12.8)
  903 FORMAT(1H0,2HNX,I5,5X,2HNY,I5,5X,7HIRSTART,I5,5X,4HIENC,I5,5X,
     1     6HNSTART,I6,5X,4HNEND,I6,5X,9HCELL SIZE,F7.2//
     2     17H DATUM LINE AT Z=,F10.3,27H WITH RESPECT TO THE ORIGIN/
     3     17H GROUND IS  AT Z=,F10.3,27H WITH RESPECT TO THE ORIGIN/
     4     17H XSHIFT IS  AT X=,F10.3,27H WITH RESPECT TO THE ORIGIN/
     5     17H YSHIFT IS  AT Y=,F10.3,27H WITH RESPECT TO THE ORIGIN/)
  904 FORMAT(1H ,7HAZIMUTH,F12.5,5X,9HELEVATION,F12.5,5X,
     1     13HBACK OFF DIST,F12.5)
  905 FORMAT(2E20.8,4E10.3)
  907 FORMAT(1H0,I5,15H  CELLS SKIPPED)
  908 FORMAT(1H0,42HOPTION SET TO COMPUTE RANDOM POINT IN CELL)
  909 FORMAT(1H0,35HOPTION SET TO CHOOSE CENTER OF CELL)
C
C1     READ GRID INPUT PARAMETERS
C
      READ(5,901)NX,NY,IRSTRT,IENC,NG1ERR,NSTART,NEND,ICENTR
      READ (5,902)A,E,ENGTH,ZSHIFT,GROUND
      READ (5,902)XSHIFT,YSHIFT,CELSIZ
C
C2     INITIALIZE PARAMETERS NOT SET BY INPUT
C
      IF(IRSTRT .LE.0)IRSTRT=1
      IF(CELSIZ .LE.0.)CELSIZ=4.
      IF(NSTART.LE.0)NSTART=1
      IF(NEND.LE.NSTART)NEND=NX*NY
      IF(NG1ERR.LE.0)NG1ERR=25
C
C3     PRINT OUT INPUT PARAMETERS
C
      WRITE (6,903)NX,NY,IRSTRT, IENC,NSTART,NEND,CELSIZ,
     1     ZSHIFT,GROUND,XSHIFT,YSHIFT
      IF(IWOT.EQ.IYES)WRITE(1,905)A,E,XSHIFT,YSHIFT,ZSHIFT,CELSIZ
      WRITE (6,904)A,E,ENGTH
      IF(ICENTR.EQ.0)WRITE(6,908)
      IF(ICENTR.NE.0)WRITE(6,909)
      RADIAN=.017453292519943
      AR=A*RADIAN
      ER=E*RADIAN
      SA=SIN(AR)
      CA=COS(AR)
      SE=SIN(ER)
      CE=COS(ER)
```

FIG. 80. Source Listing, Subroutine GRID

```
C
C       PROCESS NEND-NSTART+1 CELLS
C
      KK=NSTART
    4 WB(1)=-CE*CA
      WB(2)=-CE*SA
      WB(3)=-SE
C
C5    COMPUTE ROW AND COLUMN NUMBER OF GRID CELL
C
      II=((KK-1)/NX)+1
      J=KK-(II-1)*NX
C
      CELL2=.5*CELSIZ
      V=FLOAT((NY/2)-II)*CELSIZ +CELL2
      VREF=V+CELL2
      H=FLOAT((NX/2)- J)*CELSIZ +CELL2
      HREF=H+CELL2
      IF(ICENTR.EQ.0)GOTO 5
      H=HREF
      V=VREF
      IVIH=0
      GOTO 6
C
    5 IV=RAN(-1)*10.
      IH=RAN(-1)*10.
      IVIH=10*IH+IV
C
C6    COMPUTE RANDOM POINT WITHIN GRID CELL
C
      V=V+CELSIZ *FLOAT(IV)/10.+CELSIZ /20.
      H=H+CELSIZ *FLOAT(IH)/10.+CELSIZ /20.
C
C7    CONVERT GRID PLANE COORDINATES TO COORDINATES OF TARGET
C
    6 XBS(1)=XSHIFT-V*CA*SE-H*SA
      XBS(2)=YSHIFT-V*SA*SE+H*CA
      XBS(3)=ZSHIFT+V*CE
      CALL TROPIC(WP)
      XBS(1)=XBS(1)+WP(1)*1.0E-4
      XBS(2)=XBS(2)+WP(2)*1.0E-4
      XBS(3)=XBS(3)+WP(3)*1.0E-4
C
C8    BACK OFF RAY ORIGIN FROM GRID PLANE TO ATTACK PLANE
C
      XB(1)=XBS(1)-ENGTH*WB(1)
      XB(2)=XBS(2)-ENGTH*WB(2)
      XB(3)=XBS(3)-ENGTH*WB(3)
      IF(XB(3).LE.GROUND)GOTO 40
C
C9    SAVE RAY ORIGIN AND DIRECTION COSINES OF RAY FOR LATER REFERENCE
C
      DO 20 KK1=1,3
      XS(KK1)=XB(KK1)
      WS(KK1)=WB(KK1)
   20 CONTINUE
      CALL TRACK
      IF(IERR.GE.NG1ERR)RETURN
      IF(IRAYSK.EQ.NO)GOTO 40
```

FIG. 80. (Contd.)

```
C
C10   COMPUTE RANDOM NUMBER OF CELLS (0-25) TO BE SKIPPED
C
      MSHIFT=RAN(-1)*25.
      WRITE (6,907)MSHIFT
      KK=KK+MSHIFT
   40 KK=KK+1
      IF(KK.LE.NEND)GOTO 4
      RETURN
      END
C
C
```

FIG. 80. (Concluded)

```
      SUBROUTINE TRACK
      DIMENSION XP(3),ERROR(2)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/GTRACK/D1,D2,KHIT,LMAX,TR(200),XBS(3),IRSTRT,IENC,
     1  ITR(200),CA,CE,SA,SE
      COMMON/CAL/NIR,SLOS,ANGLE,NTYPE,SSPACE,L,XS(3),WS(3),TRAVEL,
     1  SN,V,H,IVIH
      COMMON/CONTRL/ITESTG,IRAYSK,IENTLV,IVOLUM,IWOT,ITAPE8,NO,IYES
      COMMON/WALT/LIRFO,NG1ERR
      COMMON/HOYT/VREF,HREF
      COMMON/LSU/LSURF
      COMMON/CELL/CELSIZ
      COMMON/ERR/IERR0
C
  901 FORMAT(F6.1,1X,F6.1,I3,1X,F7.2,1X,F7.2,4I2,I3,1X,2I3,
     1     1X,F8.3,1X,F8.3)
  902 FORMAT(2(I4,F7.2,F7.2,F6.1,I3,F7.2),1X,2I3,1X,I4,4X,A6)
  903 FORMAT(31H NUMBER OF INTERSECTIONS.GT.200)
  904 FORMAT(//)
  905 FORMAT(1H0,16H0 ITEM IN CELL (,I4,1H,,I4,1H),5X,
     1     2HH=,F6.1,5X,2HV=,F6.1)
C
      ERROR(2)= 6H0 ITEM
      DATA ERROR(1),ERROR(2)/4H    ,4HITEM/
      I12=4096
      NASC=-1
      IR=IRSTRT
      L=1
      KHIT=0
      JCNT=0
      MSKRT=0
      MTARG=1
      MARMR=0
      MVOL =0
C
      DO 10 I=1,200
      ITR(I)=0
      TR(I)=0.
   10 CONTINUE
C
C1    SUBROUTINE G1 WILL RETURN WITH S1=DISTANCE THRU REGION IR,
C     IRPRIM=THE NEXT REGION NUMBER, XP=INTERSECT OF NEXT REGION
C
   20 CALL G1(S1,IRPRIM,XP)
      IF(IRPRIM.LT.0)RETURN
      TR(L)=S1
      KLSURF=LSURF+7
      LOC=LIRFO+IR-1
      CALL UN2(LOC,DUM,IDENT)
      IDENT=IDENT-1
C
C2    PACK SURFACE NUMBER - BODY NUMBER - NEXT REGION NUMBER
C
      ITR(L)=(KLSURF*I12+NASC)*I12+IRPRIM
      IF(NASC.LE.NRPP)IRPRIM=0
```

FIG. 81. Source Listing, Subroutine TRACK

```
      IF(IRPRIM.EQ.0)GOTO 100
      IR=IRPRIM
      KHIT=KHIT+1
      IF(L.GT.1)GOTO 40
C
C3    COMPUTE DISTANCE FROM GRID PLANE TO FIRST INTERSECT OF TARGET
C
      D1=-((XP(1)-XBS(1))*WS(1) + (XP(2)-XBS(2))*WS(2)
     1   + (XP(3)-XBS(3))*WS(3))
      GOTO 60
C
C4    TEST SPACE IDENTIFICATION CODE
C          0 = NO SPECIAL MATERIAL  10=SKIRT  20=ARMOR  30=TARGET
C          -1,2-9,11-19,21-29,......,91-99 = INTERIOR VOLUME
C          1 = EXTERIOR VOLUME
C
   40 IF(IDENT.EQ.0)GOTO 60
      IF(IDENT-(IDENT/10)*10.EQ.0)GOTO 50
      KHIT=KHIT-1
      IF(IDENT.NE.1)MVOL=1
      GOTO 60
C
   50 IF(IDENT.EQ.20)MARMR=1
      IF(IDENT.EQ.30)MTARG=1
      IF(IDENT.EQ.10)MSKRT=1
   60 L=L+1
      IF(L.LE.200)GOTO 20
      WRITE (6,903)
      STOP
C
C5    END OF RAY   OUTPUT RESULTS
C
  100 IF(L.EQ.1)RETURN
      IF(ITAPE8.EQ.NO.AND.IWOT.EQ.NO)RETURN
C
C6    COMPUTE DISTANCE FROM GRID PLANE TO LAST INTERSECT OF TARGET
C
      D2=XDIST(XBS,XP)-S1
      D2=-D2
      IF(KHIT.GT.0)GOTO 105
      KHIT=KHIT+1
      MTARG=0
  105 KHIT=KHIT-1
      IH=ABS(H/CELSIZ )+.5
      IF(H.LT.0.)IH=-IH
      IV=ABS(V/CELSIZ )+.5
      IF(V.LT.0.)IV=-IV
C
C7    OUTPUT GRID CELL AND TARGET IDENTIFICATION DATA
C
      IF(ITAPE8.EQ.NO)GOTO 110
      WRITE (6,904)
      WRITE (6,901)HREF,VREF,IVIH,D1,D2,MSKRT,MTARG,MARMR,MVOL,
     1    KHIT,IH,IV,H,V
  110 IF(IWOT.EQ.IYES)WRITE(1,901)HREF,VREF,IVIH,D1,D2,MSKRT,MTARG,
     1    MARMR,MVOL,KHIT,IH,IV,H,V
```

FIG. 81. (Contd.)

```
C
C8     OUTPUT RAY INTERSECTION DATA
C
       LMAX=L
       L=0
       TRAVEL=TR(1)
       DO 200 KIK=1,LMAX,2
       JERR0=1
       L=L+1
       IF(L.GE.LMAX)RETURN
C
C9     COMPUTE DATA OUTPUT FOR FIRST HALF OF LINE
C
       CALL CALC
       IF(NIR.NE.0)GOTO 113
       JERR0=2
       IERR0=IERR0+1
  113  IF(SSPACE.NE.0.)JCNT=JCNT+1
C
C10    SAVE DATA OUTPUT FOR FIRST HALF OF LINE
C
       NIR1=NIR
       SLOS1=SLOS
       ANGLE1=ANGLE
       SN1=SN
       NTYPE1=NTYPE
       SPACE1=SSPACE
C
       L=L+1
       IF(L.LT.LMAX)GOTO 115
       NIR=0
       SLOS=0.
       ANGLE=0.
       SN=0.
       NTYPE=0
       SSPACE=0.
       GOTO 120
C
C11    COMPUTE DATE OUTPUT FOR SECOND HALF OF LINE
C
  115  CALL CALC
       IF(NIR.NE.0)GOTO 117
       JERR0=2
       IERR0=IERR0+1
  117  IF(SSPACE.EQ.0.)GOTO 130
  120  JCNT=JCNT+1
  130  N=L-JCNT
C
C12    TEST TRACK FLAG
C           501 = TRACK EDGE     502 = TRACK FACE
C           IF NORMAL DISTANCE    10 INCHES RAY ENTERS TRACK FACE
C
       IF(NIR1.NE.501)GOTO 140
       IF(SN1.LT.10.)NIR1=502
  140  IF(NIR.NE.501)GOTO 150
       IF(SN  .LT.10.)NIR=502
C
C13    OUTPUT RAY INTERSECTION DATA
C
```

FIG. 81. (Contd.)

```
  150 IF(IWOT.EQ.IYES)WRITE(1,902)NIR1,SLOS1,SN1,ANGLE1,NTYPE1,SPACE1,
     1      NIR,SLOS,SN,ANGLE,NTYPE,SSPACE,IH,IV,N
      IF(ITAPE8.EQ.IYES)WRITE(6,902)NIR1,SLOS1,SN1,ANGLE1,NTYPE1,SPACE1,
     1      NIR,SLOS,SN,ANGLE,NTYPE,SSPACE,IH,IV,N,ERROR(JERR0)
      IF(ITAPE8.EQ.NO.AND.JERR0.EQ.2)WRITE(6,905)IH,IV,HREF,VREF
C
      IF(L.GE.LMAX)RETURN
      IF(NTYPE .EQ.9)RETURN
  200 CONTINUE
      RETURN
      END
C
C
```

FIG. 81. (Concluded)

```
      SUBROUTINE CALC
      DIMENSION XP(3),TEMP(3),TEMP1(3),TEM(3),TEM1(3),XMID(3),IEMP(4),
     1 WN(3),WI(3),WA(3),XI(3),HF(3),VF(3)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON ASTER(10000)
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1 LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/GTRACK/D1,D2,KHIT,LMAX,TR(200),XBS(3),IRSTRT,IENC,
     1 ITR(200),CA,CE,SA,SE
      COMMON/CAL/NIR,SLOS,ANGLE,NTYPE,SSPACE,L,XS(3),WS(3),TRAVEL,
     1 SN,V,H,IVIH
      COMMON/WALT/LIRFO,NG1ERR
      EQUIVALENCE(MASTER,ASTER)
      REAL NF(3)
C
  901 FORMAT(1H0,15HTHATS ALL FOLKS//)
  902 FORMAT(1H0,17HBAD ITYPE IN CALC,5X,6HITYPE=,I5,4HNBO=,I5/
     1 16H RETURN TO TRACK//)
  903 FORMAT(1H0,23HARS DID NOT FIND NORMAL)
  904 FORMAT(//5H NORM/5H NIR=,I10,5X,6HITYPE=,I10,5X,4HNBO=,I10,5X,
     1 6HLSURF=,I10/4H WB=,3E20.10/4H WS=,3E20.10/4H XP=,3E20.8/
     2 4H XB=,3E20.10/4H XI=,3E20.10/6H XNOS=,3E20.10)
  905 FORMAT(35H ERROR IN CALC     A TRC HAS R1 = R2 )
  906 FORMAT(42H ERROR IN CALC     BAD LSURF FOR BOX OR RAW )
C
C1    RETRIEVE FOR PRESENT INTERSECT THE SURFACE NUMBER,
C     BODY NUMBER, AND NEXT REGION
C
      CALL OPENK(L,LSURF,NBO,NIR)
      IF(NIR.GT.0)GOTO 10
      WRITE (6,901)
      RETURN
C
C2    COMPUTE  TRAVEL - LINE-OF-SIGHT DISTANCE TO THIS REGION
C              SLOS   - LINE-OF-SIGHT DISTANCE THROUGH THIS REGION
C              XI     - COORDINATES OF INTERSECT POINT
C
   10 SLOS=TR(L+1)
      DO 20 I=1,3
      XI(I)=XS(I)+TRAVEL*WS(I)
   20 CONTINUE
      TRAVEL=TRAVEL+SLOS
      LSURF=LSURF-7
C
C3    SET THE CONSTANT MULTIPLIER OF THE DIRECTION COSINES OF NORMAL
C     TO +1 FOR ENTRY OR -1 FOR EXIT
C
      XNOS=1.
      IF(LSURF.LT.0)XNOS=-1.
C
C4    RETRIEVE BODY TYPE AND LOCATION OF DATA FOR INTERSECTED BODY
C
      LOC=LBODY+3*(NBO-1)
      CALL UN2(LOC,ITYPE,LDATA)
      LSURF=IABS(LSURF)
      ITYPE=ITYPE+1
      IF(ITYPE.GE.1.AND.ITYPE.LE.12)GOTO 30
```

FIG. 82. Source Listing, Subroutine CALC

```
      WRITE (6,902)ITYPE,NBO
      RETURN
C
C5    TRANSFER TO SPECIFIC BODY SECTION TO COMPUTE DIRECTION
C     COSINES OF NORMAL
C
C         RPP BOX SPH RCC REC TRC ELL RAW ARB TEC TOR ARS
   30 GOTO(50,100,150,200,200,300,350,400,450,500,550,600),ITYPE
C
C6    CHECK THE SPACE CODE AND ITEM CODE OF THE NEXT REGION
C
   40 CALL OPENK(L+1,DUM,DUM,NEXREG)
      ISPOT=LIRFO+NEXREG-1
      CALL UN2(ISPOT,DUM,IDENT)
      ISPOT=LIRFO+NIR-1
      CALL UN2(ISPOT,NIR,DUM)
      IDENT=IDENT-1
      IF(IDENT-(IDENT/10)*10.NE.0)GOTO 41
      NTYPE=0
      SSPACE=0.
      RETURN
   41 L=L+1
      IF(L+1.LT.LMAX)GOTO 42
      IDENT=9
      SSPACE=1.0E-4
      NTYPE=IDENT
      RETURN
   42 NTYPE=IDENT
      SSPACE=TR(L+1)
      TRAVEL=TRAVEL+SSPACE
      RETURN
C
C7    RPP SECTION FOR COMPUTING THE DIRECTION COSINES OF THE NORMAL
C
   50 WB(1)=0.0
      WB(2)=0.0
      WB(3)=0.0
      GOTO(51,52,53,54,55,56),LSURF
   51 WB(1)=XNOS
      GOTO 1000
   52 WB(1)=-XNOS
      GOTO 1000
   53 WB(2)=XNOS
      GOTO 1000
   54 WB(2)=-XNOS
      GOTO 1000
   55 WB(3)=XNOS
      GOTO 1000
   56 WB(3)=-XNOS
      GOTO 1000
```

FIG. 82. (Contd.)

```
      GOTO 1000
C
C8    BOX SECTION FOR COMPUTING THE DIRECTION COSINES OF THE NORMAL
C
  100 CONTINUE
      KCOM=LSURF-(LSURF/2)*2
      IF(KCOM.EQ.0)XNOS=-XNOS
      IF(LSURF-3)104,103,105
  103 I=1
      GOTO 110
  104 I=2
      GOTO 110
  105 IF(LSURF.LT.5)GOTO 103
      I=3
  110 CALL UN2(LDATA,IEMP(4),IEMP(1))
      LDATA=LDATA+1
      CALL UN2(LDATA,IEMP(2),IEMP(3))
      DO 115 J=1,3
      LH=IEMP(I)
      LV=IEMP(4)
      M=J-1
      IJK=LH+M
      IJK1=LV+M
      TEMP(J)=ASTER(IJK)+ASTER(IJK1)
      MK=J-1+IEMP(4)
      TEMP1(J)=ASTER(MK)
  115 CONTINUE
      CALL DCOSP(TEMP1,TEMP,WB)
      DO 120 J=1,3
      WB(J)=XNOS*WB(J)
  120 CONTINUE
      GOTO 1000
C
C9    SPH SECTION FOR COMPUTING THE DIRECTION COSINES OF THE NORMAL
C
  150 CALL UN2(LDATA,LV,DUM)
      DO 160 I=1,3
      M=I-1+LV
      TEM(I)=ASTER(M)
  160 CONTINUE
      CALL DCOSP(XI,TEM,WB)
      DO 170 I=1,3
      WB(I)=XNOS*WB(I)
  170 CONTINUE
      GOTO 1000
C
C10   RCC AND REC SECTION FOR COMPUTING DIRECTION COSINES OF NORMAL
C     FOR AN INTERSECT WITH EITHER PLANAR SURFACE
C
  200 IF(LSURF-2)202,201,210
  201 XNOS=-XNOS
  202 CALL UN2(LDATA,LV1,LV2)
      DO 203 I=1,3
      M=I-1
      IJK1=M+LV1
      IJK2=M+LV2
      TEM(I)=ASTER(IJK1)
      TEM1(I)=ASTER(IJK1)+ASTER(IJK2)
  203 CONTINUE
      CALL DCOSP(TEM,TEM1,WB)
```

FIG. 82. (Contd.)

```
      DO 204 I=1,3
      WB(I)=XNOS*WB(I)
  204 CONTINUE
      GOTO 1000
C
C11   RCC AND REC SECTION FOR PROJECTING INTERSECT ONTO HEIGHT VECTOR
C     FROM THE QUADRATIC SURFACE
C
  210 CALL UN2(LDATA,LV,LH)
      LR1=MASTER(LDATA+1)
      DO 211 J=1,3
      M=J-1
      IJK=LV+M
      TEM(J)=ASTER(IJK)
      IJK1=LH+M
      TEM1(J)=ASTER(IJK)+ASTER(IJK1)
  211 CONTINUE
      CALL DCOSP(TEM,XI,WN)
      CALL DCOSP(TEM,TEM1,WI)
      SUM=0.
      DO 212 J=1,3
      SUM=SUM+WN(J)*WI(J)
  212 CONTINUE
      DO 214 J=1,3
      XP(J)=SUM*XDIST(TEM,XI)
      XP(J)=XP(J)*WI(J)+TEM(J)
  214 CONTINUE
C
C12   TRANSFER TO REC SECTION TO COMPUTE DIRECTION COSINES OF NORMAL
C     IF AN INTERSECT ON THE QUADRATIC SURFACE OF AN REC
C
      IF(ITYPE.EQ.5)GOTO 250
C
C13   COMPUTE THE DIRECTION COSINES OF THE NORMAL IF AN INTERSECT ON
C     QUADRATIC SURFACE OF AN RCC
C
      CALL DCOSP(XI,XP,WB)
      DO 220 J=1,3
      WB(J)=XNOS*WB(J)
  220 CONTINUE
      GOTO 1000
C
C14   COMPUTE THE DIRECTION COSINES OF THE NORMAL IF AN INTERSECT ON
C     QUADRATIC SURFACE OF AN REC
C
  250 LDATA=LDATA+1
      CALL UN2(LDATA,LR1,LR2)
      DO 255 J=1,3
      M=J-1
      IJK1=M+LR1
      TEMP(J)=ASTER(IJK1)+XP(J)
      IJK2=M+LR2
      TEMP1(J)=ASTER(IJK2)+XP(J)
  255 CONTINUE
      A1=XDIST(XP,TEMP)
      A2=XDIST(XP,TEMP1)
      IF(A1.GE.A2)GOTO 260
      A3=A1
      A1=A2
      A2=A3
```

FIG. 82. (Contd.)

```
      TEMP(1)=TEMP1(1,
      TEMP(2)=TEMP1(2)
      TEMP(3)=TEMP1(3)
  260 C=SQRT(A1*A1=A2*A2)
      CALL DCOSP(XP,TEMP,WN)
      DO 265 J=1,3
      TEM(J)=XP(J)+C*WN(J)
      TEM1(J)=XP(J)-C*WN(J)
  265 CONTINUE
      CALL DCOSP(TEM,XI,WN)
      DO 270 J=1,3
      TEM(J)=2.*A1*WN(J)+TEM(J)
  270 CONTINUE
      CALL DCOSP(TEM,TEM1,WB)
      DO 275 J=1,3
      WB(J)=XNOS*WB(J)
  275 CONTINUE
      GOTO 1000
C
C15   TRC SECTION FOR COMPUTING THE DIRECTION COSINES OF THE NORMAL
C
  300 CALL UN2(LDATA,LV,LH)
      WN(1)=ASTER(LH)
      WN(2)=ASTER(LH+1)
      WN(3)=ASTER(LH+2)
      IF(LSURF.EQ.3)GOTO 310
      IF(LSURF.EQ.2)XNOS=-XNOS
      CALL UNIT(WN)
      WB(1)=XNOS*WN(1)
      WB(2)=XNOS*WN(2)
      WB(3)=XNOS*WN(3)
      GOTO 1000
  310 LDATA=LDATA+1
      CALL UN2(LDATA,LR1,LR2)
      RB=ASTER(LR1)
      RT=ASTER(LR2)
      RATIO=RB/(RB-RT)
      TEMP(1)=ASTER(LV)
      TEMP(2)=ASTER(LV+1)
      TEMP(3)=ASTER(LV+2)
      DO 320 I=1,3
      TEM(I)=TEMP(I)+RATIO*WN(I)-XI(I)
      TEM1(I)=TEMP(I)-XI(I)
  320 CONTINUE
      CALL CROSS(WA,TEM,TEM1)
      CALL CROSS(WB,WA,TEM)
      CALL UNIT(WB)
      WB(1)=XNOS*WB(1)
      WB(2)=XNOS*WB(2)
      WB(3)=XNOS*WB(3)
      GOTO 1000
```

FIG. 82. (Contd.)

```
C
C16    ELL SECTION FOR COMPUTING THE DIRECTION COSINES OF THE NORMAL
C
  350 CALL UN2(LDATA,LR1,LR2)
      LS=MASTER(LDATA+1)
      DO 352 J=1,3
      M=J-1
      IJK1=M+LR1
      IJK2=M+LR2
      TEM(J)=ASTER(IJK1)
      TEM1(J)=ASTER(IJK2)
  352 CONTINUE
      A=ASTER(LS)
      CALL DCOSP(TEM,XI,WN)
      DO 353 J=1,3
      TEM(J)=A*WN(J)+TEM(J)
  353 CONTINUE
      CALL DCOSP(TEM,TEM1,WB)
      DO 354 J=1,3
      WB(J)=XNOS*WB(J)
  354 CONTINUE
      GOTO 1000
C
C17    RAW SECTION FOR COMPUTING THE DIRECTION COSINES OF NORMAL TO
C      SLANTED SURFACE
C
  400 IF(LSURF.EQ.2)GOTO 401
C
C18    TRANSFER TO BOX SECTION IF INTERSECT NOT ON SLANT SIDE
C
      IF(LSURF.NE.4)GOTO 100
      WRITE (6,906)
      STOP
  401 CALL UN2(LDATA,LV,LV1)
      LDATA=LDATA+1
      CALL UN2(LDATA,LV2,LV3)
      DO 410 J=1,3
      M=J-1
      IJK1=M+LV1
      IJK2=M+LV2
      TEMP(J)=ASTER(IJK1)
      XMID(J)=ASTER(IJK1)-ASTER(IJK2)
      IJK3=M+LV3
      TEM(J)=ASTER(IJK3)
  410 CONTINUE
      I=1
      J=2
      K=3
      LK=0
      DO 411 KK=1,3
      TEM1(I)=XMID(J)*TEM(K)-XMID(K)*TEM(J)
      LK=I
      I=J
      J=K
      K=LK
  411 CONTINUE
```

FIG. 82. (Contd.)

```
      SUM=0.
      DO 412 J=1,3
      SUM=TEM1(J)*TEMP(J)+SUM
  412 CONTINUE
      SUM=-SUM/ABS(SUM)
      TLK=TEM1(1)**2+TEM1(2)**2+TEM1(3)**2
      TLK=SQRT(TLK)
      DO 420 J=1,3
      WB(J)=XNOS*SUM*TEM1(J)/TLK
  420 CONTINUE
      GOTO 1000
C
C19   ARB SECTION FOR COMPUTING THE DIRECTION COSINES OF THE NORMAL
C
  450 LSPT=LDATA+LSURF-1
      CALL UN2(LSPT,DUM,L1)
      SUM=0.
      DO 451 J=1,3
      M=J-1
      SUM=SUM+ASTER(IJK)**2
  451 CONTINUE
      DIV=SQRT(SUM)
      DO 460 J=1,3
      M=J-1
      IJK=M+L1
      WB(J)=XNOS*ASTER(IJK)/DIV
  460 CONTINUE
      GOTO 1000
C
C20   TEC SECTION FOR COMPUTING THE DIRECTION COSINES OF THE NORMAL
C
  500 CALL UN2(LDATA,LV,LH)
      LDATA=LDATA+1
      CALL UN2(LDATA,LN,LA)
      WN(1)=ASTER(LN)
      WN(2)=ASTER(LN+1)
      WN(3)=ASTER(LN+2)
      IF(LSURF-2)520,510,530
  510 XNOS=-XNOS
  520 WB(1)=XNOS*WN(1)
      WB(2)=XNOS*WN(2)
      WB(3)=XNOS*WN(3)
      GOTO 1000
  530 LDATA=LDATA+1
      CALL UN2(LDATA,LR1,LR2)
      LR3=MASTER(LDATA+1)
      VF(1)=ASTER(LV)
      VF(2)=ASTER(LV+1)
      VF(3)=ASTER(LV+2)
      HF(1)=ASTER(LH)
      HF(2)=ASTER(LH+1)
      HF(3)=ASTER(LH+2)
      TEMP(1)=XI(1)-VF(1)
      TEMP(2)=XI(2)-VF(2)
      TEMP(3)=XI(3)-VF(3)
      HH=DOT(TEMP,WN)
      HDN=DOT(HF,WN)
      GAMMA=HH/HDN
```

FIG. 82. (Contd.)

```
      TEMP(1)=VF(1)+GAMMA*HF(1)
      TEMP(2)=VF(2)+GAMMA*HF(2)
      TEMP(3)=VF(3)+GAMMA*HF(3)
      R1=ASTER(LR1)
      R2=ASTER(LR2)
      TAU=(R1/R2)**2
      R4=R2/ASTER(LR3)
      BSQ=(GAMMA*R4+R2*(1.-GAMMA))**2
      ASQ=TAU*BSQ
      C=SQRT(ASQ-BSQ)
      TWOA=2.*SQRT(ASQ)
      DO 540 I=1,3
      IJK=LA+I-1
      TEMP1(I)=C*ASTER(IJK)
      TEM(I)=TEMP(I)+TEMP1(I)
      TEM1(I)=TEMP(I)-TEMP1(I)
  540 CONTINUE
      CALL DCOSP(TEM,XI,WN)
      TEMP(1)=TEM(1)+TWOA*WN(1)
      TEMP(2)=TEM(2)+TWOA*WN(2)
      TEMP(3)=TEM(3)+TWOA*WN(3)
      CALL DCOSP(TEMP,TEM1,WN)
      IF(R2.EQ.R4)GOTO 545
      RATIO=R2/(R2-R4)
      HF(1)=VF(1)+RATIO*HF(1)-XI(1)
      HF(2)=VF(2)+RATIO*HF(2)-XI(2)
      HF(3)=VF(3)+RATIO*HF(3)-XI(3)
  545 CALL CROSS(NF,HF,WN)
      CALL CROSS(WB,NF,HF)
      CALL UNIT(WB)
      WB(1)=XNOS*WB(1)
      WB(2)=XNOS*WB(2)
      WB(3)=XNOS*WB(3)
      GOTO 1000
C
C21   TOR SECTION FOR COMPUTING THE DIRECTION COSINES OF THE NORMAL
C
  550 CALL UN2(LDATA,LV,LN)
      LDATA=LDATA+1
      CALL UN2(LDATA,LR1,DUM)
      DO 551 I=1,3
      J=I-1
      IJK=LV+J
      TEMP(I)=XI(I)-ASTER(IJK)
      IJK=LN+J
      TEMP1(I)=ASTER(IJK)
  551 CONTINUE
      R1=ASTER(LR1)
      CALL CROSS(TEM,TEMP1,TEMP)
      CALL CROSS(TEM1,TEM,TEMP1)
      CALL UNIT(TEM1)
```

FIG. 82. (Contd.)

```
      DO 552 I=1,3
      J=I-1
      IJK=LV+J
      TEM(I)=ASTER(IJK)
      TEMP1(I)=TEM(I)+R1*TEM1(I)
  552 CONTINUE
      CALL DCOSP(TEMP1,XI,WB)
      DO 553 I=1,3
      WB(I)=XNOS*WB(I)
  553 CONTINUE
      GOTO 1000
C
C22   ARS SECTION FOR COMPUTING THE DIRECTION COSINES OF THE NORMAL
C
  600 LOCARS=MASTER(LDATA)
      LOC=LOCARS+2
      DIS=XDIST(XS,XI)
      DO 610 I=1,20
      IF(ABS(DIS-ASTER(LOC)).LE.0.0001)GOTO 620
      LOC=LOC+4
  610 CONTINUE
      WRITE(6,903)
      SN=-1.
      ANGLE=-1.
      RETURN
  620 WB(1)=ASTER(LOC+1)
      WB(2)=ASTER(LOC+2)
      WB(3)=ASTER(LOC+3)
      GOTO 1000
C
C23   COMPUTE OBLIQUITY ANGLE AND NORMAL DISTANCE TO NEXT REGION
C
 1000 DO 1001 J=1,3
      XB(J)=XI(J)+WS(J)*1.0E-3
 1001 CONTINUE
      ANGLE=0.
      DO 1002 J=1,3
      ANGLE=ANGLE+WB(J)*WS(J)
 1002 CONTINUE
      IF(ABS(ANGLE).LE.1.)GOTO 1010
      ANGLE=0.
      SN=0.
      WRITE (6,904)NIR,ITYPE,NBO,LSURF,WB,WS,XP,XB,XI,XNOS
      IR=NIR
      GOTO 40
C
C24   COMPUTE OBLIQUITY ANGLE
C
 1010 ANGLE=ATAN2(SQRT(1.-ANGLE*ANGLE),ANGLE)*180./3.141592654
      IF(ANGLE.LE.90.)GOTO 1020
      DO 1011 J=1,3
      WB(J)=-WB(J)
 1011 CONTINUE
      GOTO 1000
```

FIG. 82. (Contd.)

```
C
C25    COMPUTE NORMAL DISTANCE TO NEXT INTERSECT
C
 1020 NASC=-2
      IR=NIR
      CALL G1(S1,IRPRIM,XP)
      SN=S1
      GOTO 40
      END
C
C
```

FIG. 82. (Concluded)

```
      SUBROUTINE G1(S1,IRPRIM,XP)
      DIMENSION XP(3),LSURT(50),NASCT(50)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGO,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/CAL/NIR,SLOS,ANGLE,NTYPE,SSPACE,L,XS(3),WS(3),TRAVEL,
     1  SN,V,H,IVIH
      COMMON/WALT/LIRFO,NG1ERR
      COMMON/LSU/LSURF
      COMMON/CONTRL/ITESTG,IRAYSK,IENTLV,IVOLUM,IWOT,ITAPE8,NO,IYES
      COMMON/DAVIS/IGRID,LOOP,INORM
      COMMON/CELL/CELSIZ
      COMMON/WHICH/NBO
      EQUIVALENCE (ASTER,MASTER)
C
  901 FORMAT(1H0,32HERROR IN G1 AT 140      BAD ITYPE,5X,4HITY=,I5)
  902 FORMAT(1H0,33HERROR IN G1 AT 510      SM %= PINF,5X,3HIR=,I5)
  903 FORMAT(4H XB=,3E20.8/4H WB=,3E20.8/10X,5HKLOOP,12X,3HNBO,
     1  12X,3HLRI,12X,3HLRO,11X,4HNHIT,11X,4HLOOP/6I15)
  904 FORMAT(1H1,15(2H* ),3X, 9HERROR NO.,I5,3X,15(2H *)//)
  905 FORMAT(34X,4HCELL,2I4)
  906 FORMAT(19H ERROR IN G1 AT 640//4H J1=,I10,4H J2=,I10,7H LSURF=,
     1  I10,6H NASC=,I10,4H IR=,I10/4H SM=,E21.10,4H S1=,E17.10/
     2  4H WB=,3E21.10/4H XB=,3E21.10)
  907 FORMAT(50H THE (SOLID POSITION/DEPTH/POINT NOW AT) IS ONE OF,
     1  6H THESE/6H XBD =,3E21.10/6H DIST=,E21.10//)
  908 FORMAT(9X,3HRIN,12X,4HROUT,7X,8HENTERING,2X,7HLEAVING,3X,
     1  8HBODY NO.,5X,3HRAY,/35X,8HSIDE NO.,2X,8HSIDE NO.//)
  910 FORMAT(//16H TILT  RIN=ROUT=,E20.10,30X,2HI=,I5//)
  911 FORMAT(2(2X,E15.8),4X,I2,8X,I2,6X,I5,5X,7HSTARTED/)
  912 FORMAT(2(2X,E15.8),4X,I2,8X,I2,6X,I5,5X,7HHAS HIT/)
  913 FORMAT(2(2X,E15.8),4X,I2,8X,I2,6X,I5,5X,7HLEAVING/)
  914 FORMAT(2(2X,E15.8),4X,I2,8X,I2,6X,I5,5X,7H IN     /)
  915 FORMAT(2(2X,E15.8),4X,I2,8X,I2,6X,I5,5X,8HENTERING/)
  916 FORMAT(2(2X,E15.8),4X,I2,8X,I2,6X,I5,5X,8HWILL HIT/)
  917 FORMAT(//4(14H END ERROR NO.,I4,3X)/)
  918 FORMAT(1H0,I5,21H ERRORS IN G1, RETURN)
C
      INORM=0
      IF(NASC.EQ.-2)INORM=1
      S1=0.
      IF(NASC.GT.0)GOTO 20
C
C1    INITIALIZE FOR NEW RAY
C
      DIST=0.
      IF(KLOOP.LT.32000)GOTO 15
      KLOOP=0
      LION=LIO+NBODY+NRPP-1
      DO 10 I=LIO,LION
      MASTER(I)=0
   10 CONTINUE
   15 KLOOP=KLOOP+1
C
C2    BEGIN/CONTINUE TRACING RAY THRU REGION
C
   20 SM=PINF
      NHIT=0
```

FIG. 83. Source Listing, Subroutine G1

```
C
C3     COMPUTE LOCATION OF REGION DATA
C
       LOC=LREGD+IR-1
C
C4     RETRIEVE THE NUMBER OF BODIES IN REGION
C
       CALL UN2(LOC,LOC,NC)
       LOC=LOC-1
C
       DO 500 N=1,NC
C
C5     RETRIEVE BODY NUMBER
C
       LOC=LOC+1
       CALL UN2(LOC,DUM,NBO)
C
C6     RETRIEVE ENTER AND EXIT SURFACE NUMBERS AND LAST RAY NUMBER
C
       ITEMP=LIO+NBO-1
       CALL UN3(ITEMP,LRI,LRO,LOOP)
C
C7     RETRIEVE BODY TYPE AND LOCATION OF DATA
C
       ITEMP=LBODY+3*(NBO-1)
       CALL UN2(ITEMP,ITYPE,LOCDA)
       IF(LOOP.NE.KLOOP)GOTO 130
C
C8     CONTINUE RAY   RETRIEVE RIN/ROUT FOR CURRENT BODY
C
       IF(ITYPE.GT.11)GOTO 140
       IJK=LRIN+NBO-1
       RIN=ASTER(IJK)
       IJK=LROT+NBO-1
       ROUT=ASTER(IJK)
       IF(ITYPE.LT.10)GOTO 320
C
C9     IS NEXT RIN/ROUT SET REQUIRED FOR TOR OR ARS
C
       IF(ROUT.LT.0.)GOTO 320
       IF(DIST.LT.ROUT)GOTO 320
       IF(NASC.EQ.NBO)NASC=0
C
  130 LRI=1
       LRO=1
       ITY=ITYPE+1
       IF(ITY.GE.1.AND.ITY.LE.12)GOTO 200
  140 IERR=IERR+1
       WRITE (6,901)ITYPE
       GOTO 800
C
C10    COMPUTE RIN/ROUT FOR CURRENT BODY
C
C            RPP BOX SPH RCC REC TRC ELL RAW ARB TEC TOR ARS
  200 GOTO(205,210,215,220,225,230,235,240,245,250,255,260),ITY
  205 CALL RPP(NBO)
       GOTO 300
  210 CALL BOX
       GOTO 300
  215 CALL SPH
       GOTO 300
```

FIG. 83. (Contd.)

```
  220 CALL RCC
      GOTO 300
  225 CALL REC
      GOTO 300
  230 CALL TRC
      GOTO 300
  235 CALL ELL
      GOTO 300
  240 CALL RAW
      GOTO 300
  245 CALL ARB
      GOTO 300
  250 CALL TEC
      GOTO 300
  255 CALL TOR
      GOTO 300
  260 CALL ARS
C
C11   STORE RIN AND ROUT FOR BODY IN RIN AND ROUT TABLES
C
  300 IJK=LRIN+NBO-1
      ASTER(IJK)=RIN
      IJK=LROT+NBO-1
      ASTER(IJK)=ROUT
      IJK=LIO+NBO-1
      MASTER(IJK)=KLOOP+I15*(LRO+64*LRI)
C
C12   IS POINT XP ON CURRENT BODY   YES=IS IT ENTER OR EXIT
C
  320 IF(NASC.NE.NBO)GOTO 330
      IF(LSURF)500,500,340
C
C13   DOES RAY INTERSECT BODY   YES=DOES IT ORIGINATE WITHIN BODY
C
  330 IF(ROUT.LE.0.)GOTO 500
      IF(RIN.GT.0.)GOTO 350
C
C14   POINT XP AT RIN OR WITHIN BODY
C
  340 IF(ABS(ROUT-SM).GT.SM*1.0E-6)GOTO 341
      ROUT=SM
      IJK=LROT+NBO-1
      ASTER(IJK)=ROUT
      GOTO 345
  341 IF(ROUT-SM)342,345,500
  342 IF(DIST.GE.ROUT)GOTO 500
      NHIT=0
  345 NHIT=NHIT+1
      SM=ROUT
      LSURT(NHIT)=-LRO
      NASCT(NHIT)=NBO
      GOTO 500
C
C15   POINT XP AT ROUT OF BODY
C
  350 IF(ABS(RIN-SM).GT.SM*1.0E-6)GOTO 351
      RIN=SM
      IJK=LRIN+NBO-1
      ASTER(IJK)=RIN
      GOTO 355
```

FIG. 83. (Contd.)

```
  351 IF(RIN-SM)352,355,500
  352 IF(DIST.GE.RIN)GOTO 340
      NHIT=0
  355 NHIT=NHIT+1
      SM=RIN
      LSURT(NHIT)=LRI
      NASCT(NHIT)=NBO
  500 CONTINUE
C
      IF(SM.LT.PINF)GOTO 530
C
C16    ERROR-NO INTERSECT
C
      WRITE (6,902)IR
      WRITE (6,903)XB,WB,KLOOP,NBO,LRI,LRO,NHIT,LOOP
      GOTO 700
C
C17    COMPUTE NEW COORDINATES OF POINT XP AND REVISE DISTANCE TRAVELLED
C
  530 SI=SI+SM-DIST
      DIST=SM
      XP(1)=XB(1)+SM*WB(1)
      XP(2)=XB(2)+SM*WB(2)
      XP(3)=XB(3)+SM*WB(3)
C
      IF(NASC.EQ.-2)RETURN
C
C18    DETERMINE REGION THAT POINT XP NOW IN
C
      DO 640 NN=1,NHIT
      NASC =NASCT(NN)
      LSURF=LSURT(NN)
      LTRUE=0
C
C19    COMPUTE LOCATION OF INTERSECTED BODY DATA
C
      LOC=LBODY+3*(NASC-1)
      LOC=LOC+1
C
C20    RETRIEVE LOCATIONS OF REGION ENTER/LEAVE TABLE FOR BODY
C
      CALL UN2(LOC,LENT,LEAV)
      LOC=LOC+1
C
C21    RETRIEVE NUMBER OF REGIONS IN ENTRY LIST AND EXIT LIST
C
      CALL UN2(LOC,NENT,NEAV)
C
C22    COMPUTE THE BEGIN AND END OF LIST
C
      IF(LSURF.LE.0)GOTO 600
      J1=LENT
      J2=LENT+NENT-1
      GOTO 610
  600 J1=LEAV
      J2=LEAV+NEAV-1
C
C23    ANY REGIONS IN LIST OR IS RAY LEAVING RPP
C
  610 IRPRIM=MASTER(J2)
```

FIG. 83. (Contd.)

```
      IF(J1.LE.J2)GOTO 620
      IF(NASC.GT.NRPP)GOTO 700
      IF(LSURF)630,700,700
C
C24   DETERMINE REGION POINT XP NOW ENTERING
C
  620 DO 625 J=J1,J2
      IRPRIM=MASTER(J)
      CALL WOWI(IRPRIM,LSURF,NASC,LTRUE)
      IF(LTRUE.GT.0)GOTO 650
  625 CONTINUE
C
C25   RAY LEAVING RPP
C
      IF(NASC.GT.NRPP)GOTO 640
      IF(LSURF)630,700,640
  630 CALL RPP2(LSURF,XP,IRP)
      IF(IRP.GT.0)GOTO 631
      IRPRIM=0
      RETURN
C
C26   RETRIEVE LOCATION/NUMBER OF REGION ENTER LIST
C     COMPUTE BEGINNING AND END OF LIST
C
  631 LTRUE=0
      LOC=LBODY+3*(IRP-1)
      LOC=LOC+1
      CALL UN2(LOC,LENT,LEAV)
      LOC=LOC+1
      CALL UN2(LOC,NENT,NEAV)
      J1=LENT
      J2=LENT+NENT-1
      IF(J1.GT.J2)GOTO 700
C
C27   DETERMINE REGION POINT XP NOW ENTERING IN NEW RPP
C
      DO 632 J=J1,J2
      IRPRIM=MASTER(J)
      CALL WOWI(IRPRIM,LSURF,IRP,LTRUE)
      IF(LTRUE.GT.0)GOTO 650
  632 CONTINUE
  640 CONTINUE
      GOTO 700
C
C28   REGION POINT XP ENTERING HAS BEEN DETERMINED
C
  650 IF(IR.EQ.IRPRIM)GOTO 660
      IF(S1.EQ.0.)GOTO 660
      IF(S1.LT.0.)GOTO 700
      IF(ABS(S1).LE.1.0E-6)GOTO 660
      IF(IVOLUM.EQ.IYES)RETURN
      IF(ITESTG.EQ.IYES)RETURN
C
C29   RETRIEVE SPACE AND COMPONENT CODE OF REGION
C
      LOC=LIRFO+IR-1
      CALL UN2(LOC,ICODE,IDENT)
      LOC=LIRFO+IRPRIM-1
      CALL UN2(LOC,ICODE1,IDENT1)
      IF(IDENT.EQ.1)GOTO 655
```

FIG. 83. (Contd.)

```
      IF(IDENT.EQ.IDENT1)GOTO 660
      RETURN
  655 IF(ICODE.NE.ICODE1)RETURN
  660 IR=IRPRIM
      GOTO 20
C
C30   START OF ERROR DIAGNOSTIC SECTION
C
  700 IERR=IERR+1
      WRITE (6,904)IERR
C
C31   COMPUTE GRID CELL NUMBER IF G1 NOT CALLED BY VOLUM OR TESTG
C
      IF(IVOLUM.EQ.IYES.OR.ITESTG.EQ.IYES)GOTO 705
      IH=ABS(H/CELSIZ )+.5
      IF(H.LT.0.)IH=-IH
      IV=ABS(V/CELSIZ )+.5
      IF(V.LT.0.)IV=-IV
      WRITE (6,905)IH,IV
  705 WRITE (6,906)J1,J2,LSURF,NASC,IR,SM,S1,WB,XB
C
C32   COMPUTE COORDINATES OF XP AT TIME OF ERROR
C
      XBD(1)=XB(1)-DIST
      XBD(2)=XB(2)-DIST
      XBD(3)=XB(3)-DIST
      WRITE (6,907)XBD,DIST
      WRITE (6,908)
      NN=NBODY+NRPP
C
C33   PRINT OUT PERTINENT DATA FOR ALL BODIES IN REGION INTERSECTED
C     BY RAY FOR ERROR ANALYSIS
C
      DO 750 I=1,NN
      LOC=LIO+I-1
      CALL UN3(LOC,I1,I2,I3)
      IF(KLOOP.NE.I3)GOTO 750
      IJK=LRIN+I-1
      RIN=ASTER(IJK)
      IJK=LROT+I-1
      ROUT=ASTER(IJK)
      IF(RIN.NE.ROUT)GOTO 710
      WRITE (6,910)RIN,I
      GOTO 750
C
  710 IF(ABS(RIN).NE.PINF)GOTO 720
      IF(ABS(ROUT)-PINF)740,750,740
  720 IF(RIN-DIST)730,744,745
  730 IF(ROUT-DIST)741,742,743
C
  740 WRITE (6,911)RIN,ROUT,I1,I2,I
      GOTO 750
  741 WRITE (6,912)RIN,ROUT,I1,I2,I
      GOTO 750
  742 WRITE (6,913)RIN,ROUT,I1,I2,I
      GOTO 750
  743 WRITE (6,914)RIN,ROUT,I1,I2,I
      GOTO 750
  744 WRITE (6,915)RIN,ROUT,I1,I2,I
      GOTO 750
```

FIG. 83. (Contd.)

```
  745 WRITE (6,916)RIN,ROUT,I1,I2,I
C
  750 CONTINUE
      WRITE (6,917)IERR,IERR,IERR,IERR
      IRPRIM=-1
C
  800 IF(IERR.GE.NG1ERR)WRITE (6,918)NG1ERR
      RETURN
      END
C
C
```

FIG. 83. (Concluded)

```
      SUBROUTINE WOWI(JREG,LSURF,NEX,LTRUE)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/DAVIS/IGRID,LOOP,INDRM
      COMMON/WHICH/NBO
      EQUIVALENCE(ASTER,MASTER)
C
  901 FORMAT(1H0,32HERROR IN G1 AT 140     BAD ITYPE,5X,4HITY=,I5)
C
      LOC=LREGD+JREG-1
C
C1    RETRIEVE NUMBER OF BODIES IN REGION AND LOCATION OF
C     OPERATOR/BODY LIST
C
      CALL UN2(LOC,LOCD,NC)
C
C2    RETRIEVE FIRST OPERATOR/BODY FROM LIST
C
      CALL UN2(LOCD,IOP,NBO)
      N=1
      IOPER=IOP
C
C3    RETRIEVE ENTER AND EXIT SURFACE NUMBERS AND NUMBER OF LAST RAY
C
   10 ITEMP=LIO+NBO-1
      CALL UN3(ITEMP,LRI,LRO,LOOP)
C
C4    RETRIEVE BODY TYPE AND LOCATION OF DATA
C
      ITEMP=LBODY+3*(NBO-1)
      CALL UN2(ITEMP,ITYPE,LOCDA)
      IF(LOOP.NE.KLOOP)GOTO 30
C
      IF(ITYPE.GT.11)GOTO 40
C
C5    RETRIEVE RIN AND ROUT FOR CURRENT BODY
C
      IJK=LRIN+NBO-1
      RIN=ASTER(IJK)
      IJK=LROT+NBO-1
      ROUT=ASTER(IJK)
      IF(ITYPE.LT.10)GOTO 310
C
C6    IS NEXT RIN/ROUT SET REQUIRED FOR TOR OR ARS
C
      IF(ROUT.LT.0.)GOTO 400
      IF(DIST.LE.ROUT)GOTO 310
C
   30 LRI=1
      LRO=1
      ITY=ITYPE+1
      IF(ITY.GE.1.AND.ITY.LE.12)GOTO 100
   40 IERR=IERR+1
      WRITE (6,901)ITYPE
      RETURN
```

FIG. 84. Source Listing, Subroutine WOWI

```
C
C7      COMPUTE RIN/ROUT FOR CURRENT BODY
C
C             RPP BOX SPH RCC REC TRC ELL RAN ARB TEC TOR ARS
  100 GOTO(110,120,130,140,150,160,170,180,190,200,210,220),ITY
  110 CALL RPP(NBO)
      GOTO 300
  120 CALL BOX
      GOTO 300
  130 CALL SPH
      GOTO 300
  140 CALL RCC
      GOTO 300
  150 CALL REC
      GOTO 300
  160 CALL TRC
      GOTO 300
  170 CALL ELL
      GOTO 300
  180 CALL RAW
      GOTO 300
  190 CALL ARB
      GOTO 300
  200 CALL TEC
      GOTO 300
  210 CALL TOR
      GOTO 300
  220 CALL ARS
C
  300 IJK=LIO+NBO-1
      MASTER(IJK)=KLOOP+I15*(LRO+64*LRI)
C
C8      DETERMINE CORRECT RIN/ROUT AND STORE IN ASTER ARRAY
C
  310 IF(ROUT.LE.0.)GOTO 330
      IF(ABS(RIN-DIST).GT.DIST*1.0E-6)GOTO 320
      RIN=DIST
      GOTO 330
C
  320 IF(ABS(ROUT-DIST).LE.DIST*1.0E-6)ROUT=DIST
C
  330 IJK=LRIN+NBO-1
      ASTER(IJK)=RIN
      IJK=LROT+NBO-1
      ASTER(IJK)=ROUT
C
C9      TEST CONDITIONS FOR POINT XB IN REGION UNDER TEST
C
  400 IF(IOPER.GT.4)GOTO 500
C
C10     (+) OPERATOR TEST   RIN.LE.DIST.LT.ROUT   POINT XB IN BODY
C
      IF(RIN.GT.DIST)GOTO 700
      IF(DIST-ROUT)600,700,700
C
C11     (-) OPERATOR TEST   ROUT.LE.0 OR DIST.LT.RIN OR DIST.GE.ROUT
C       POINT XB OUTSIDE OF BODY
C
  500 IF(ROUT.LE.0.)GOTO 600
      IF(DIST.LT.RIN)GOTO 600
      IF(DIST.EQ.RIN)GOTO 700
```

FIG. 84. (Contd.)

```
      IF(DIST.LT.ROUT)GOTO 700
C
C12   CHECK NEXT BODY IN OPERATOR/BODY LIST
C
  600 IF(N.GE.NC)GOTO 800
      N=N+1
      LOCD=LOCD+1
      CALL UN2(LOCD,IOPER,NBO)
      IF(IOPER.EQ.1.OR.IOPER.EQ.5)GOTO 800
      GOTO 10
C
C13   (OR) OPERATOR TEST
C     ALL (+) OR (-) IN (OR) SERIES MUST BE VALID
C
  700 IF(IOP.NE.1.AND.IOP.NE.5) RETURN
      IF(N.GE.NC)RETURN
      N=N+1
      DO 710 NN=N,NC
      LOCD=LOCD+1
      CALL UN2(LOCD,IOPER,NBO)
      IF(IOPER.EQ.1.OR.IOPER.EQ.5)GOTO 720
  710 CONTINUE
      RETURN
  720 N=NN
      GOTO 10
C
C14   POINT XB WITHIN CURRENT REGION. LTRUE = 1
C
  800 LTRUE=LTRUE+1
      RETURN
      END
C
C
```

FIG. 84. (Concluded)

```
      SUBROUTINE RPP(NBO)
      DIMENSION PR(6),LR(6),XS(6),LST(6)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGO,
     1 LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      EQUIVALENCE (MASTER,ASTER)
C
  901 FORMAT(1H0,12HERROR IN RPP/4H L =,I10,5X,4HNBO=,I10,5X,3HIR=,
     1 I10/4H XB=,3E20.10/4H WB=,3E20.10/4H PR=,6E20.10/4H LR=,6I10)
C
C1    SET UP SIX MEMBER ARRAY TO REPRESENT COORDINATE PAIRS
C
      LST(1)=1
      LST(2)=1
      LST(3)=2
      LST(4)=2
      LST(5)=3
      LST(6)=3
      L=0
      PR(1)=0.
      PR(2)=0.
C
C2    RETRIEVE THE SIX BOUNDARIES OF THE RPP
C
      DO 10 I=1,6
      XS(I)=S(NBO,I)
   10 CONTINUE
C
      DO 100 I=1,6
      II=LST(I)
      TEMP=XS(I)-XB(II)
      IF(WB(II)) 20,100,30
   20 IF(TEMP)40,100,100
   30 IF(TEMP.LE.0.)GOTO 100
   40 TRY=TEMP/WB(II)
      DO 60 J=1,3
      IF(J.EQ.II)GOTO 60
C
C3    COMPUTE INTERSECT/PLANE COORDINATE
C
      XRY=XB(J)+TRY*WB(J)
C
C4    DETERMINE IF INTERSECT OCCURS WITHIN BOUNDARY OF PLANE
C
      IF((XS(2*J-1)-XRY)*(XRY-XS(2*J)).LT.0.)GOTO 100
   60 CONTINUE
      L=L+1
C
C5    COMPUTE DISTANCE TO INTERSECT POINT
C
      PR(L)=TRY
      LR(L)=I
      IF(L.EQ.2)GOTO 130
      IF(L.LT.2)GOTO 100
      WRITE (6,901)L,NBO,IR,XB,WB,PR,LR
      ROUT=-PINF
      RETURN
  100 CONTINUE
```

FIG. 85. Source Listing, Subroutine RPP

```
      GOTO 160
C
  130 IF(ABS(PR(1)-PR(2)).LE.PR(1)*1.0E-6)GOTO 200
      IF(PR(1)-PR(2))140,180,150
C
C6    COMPUTE RIN, ROUT, AND SURFACE NUMBERS OF INTERSECTS
C
  140 RIN=PR(1)
      LRI=LR(1)
      ROUT=PR(2)
      LRO=LR(2)
      RETURN
  150 RIN=PR(2)
      LRI=LR(2)
      ROUT=PR(1)
      LRO=LR(1)
      RETURN
C
  160 IF(L.GE.1)GOTO 180
C
C7    ASSIGN VALUE TO ROUT FOR NO INTERSECTION
C
  170 ROUT=-PINF
      RETURN
C
C8    RAY ORIGINATES WITHIN RPP
C
  180 RIN=-PINF
      LRI=0
      ROUT=PR(1)
      LRO=LR(1)
      RETURN
C
C9    DETERMINE IF RAY ORIGINATES WITHIN RPP OR MISSES
C
  200 DO 220 J=1,3
      IF(XB(J).LT.XS(2*J-1))GOTO 170
      IF(XB(J).GT.XS(2*J))GOTO 170
  220 CONTINUE
      GOTO 180
      END
C
C
```

FIG. 85. (Concluded)

```
      SUBROUTINE BOX
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      EQUIVALENCE (MASTER,ASTER)
C
C1    RETRIEVE LOCATION OF BOX VERTEX AND H1 COORDINATES
C
      CALL UN2(LOCDA,IV,IH1)
      LOC=LOCDA+1
C
C2    RETRIEVE LOCATION OF BOX H2 AND H3 COORDINATES
C
      CALL UN2(LOC,IH2,IH3)
      RIN=-PINF
      ROUT=PINF
      DO 105 I=1,3
      IF(I-2)11,12,13
   11 II=2
      GOTO 14
   12 II=1
      GOTO 14
   13 II=3
   14 A=0.
      VP=0.
      W=0.
C
C3    COMPUTE VECTOR DOT PRODUCTS
C
      DO 15 J=1,3
      JV=IV+J
      JA=IH1+J
      VP=VP+(ASTER(JV-1)-XB(J))*ASTER(JA-1)
      W=W+WB(J)*ASTER(JA-1)
      A=A+ASTER(JA-1)**2
   15 CONTINUE
      IF(W)30,20,40
   20 IF(-VP.LT.0.)GOTO 200
      IF(-VP-A)100,100,200
C
C4    COMPUTE ROUT
C
   30 CP=VP/W
      LO=2*II-1
      IF(CP.LE.0.)GOTO 200
C
C5    COMPUTE RIN
C
      CM=(VP+A)/W
      LI=LO+1
      GOTO 60
C
C6    COMPUTE ROUT
C
   40 CP=(VP+A)/W
      LO=2*II
      IF(CP.LE.0.)GOTO 200
```

FIG. 86. Source Listing, Subroutine BOX

```
C
C7      COMPUTE RIN
C
        CM=VP/W
        LI=LO-1
   60   IF(ROUT.LE.CP)GOTO 80
        ROUT=CP
        LRO=LO
   80   IF(RIN.GE.CM)GOTO 100
        RIN=CM
        LRI=LI
  100   IH1=IH2
        IH2=IH3
  105   CONTINUE
        IF(ABS(RIN-ROUT).LE.ROUT*1.0E-6)GOTO 200
        IF(RIN.LT.ROUT)RETURN
C
  200   RIN=PINF
        ROUT=-PINF
        RETURN
        END
C
C
```

FIG. 86. (Concluded)

```
      SUBROUTINE SPH
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
C
C1    RETRIEVE LOCATION OF SPH VERTEX AND RADIUS
C
      CALL UN2(LOCDA,ITEMP,I2)
      R=ASTER(I2)
      ITEMP=ITEMP+1
      DX=XB(1)-ASTER(ITEMP-1)
      DY=XB(2)-ASTER(ITEMP)
      DZ=XB(3)-ASTER(ITEMP+1)
      B=DX*WB(1)+DY*WB(2)+DZ*WB(3)
      C=DX*DX+DY*DY+DZ*DZ-R*R
      DIS=B*B-C
      IF(C.GT.0.)GOTO 10
C
C2    RAY ORIGINATES WITHIN SPHERE
C
      RIN=-PINF
      ROUT=SQRT(DIS)-B
      RETURN
C
   10 IF(DIS.GT.0.)GOTO 20
C
C3    RAY MISSES SPHERE
C
      RIN=PINF
      ROUT=-PINF
      RETURN
C
C4    RAY INTERSECTS SPHERE
C
   20 DIS=SQRT(DIS)
      RIN=-B-DIS
      ROUT=-B+DIS
      RETURN
      END
C
C
```

FIG. 87. Source Listing, Subroutine SPH

```
      SUBROUTINE RCC
      DIMENSION V(3),H(3)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      EQUIVALENCE (ASTER,MASTER)
C
C1    RETRIEVE LOCATION OF RCC VERTEX AND HEIGHT VECTOR COORDINATES
C
      CALL UN2(LOCDA,IV,IH)
C
C2    RETRIEVE LOCATION OF RADIUS
C
      IRR=MASTER(LOCDA+1)
C
C3    RETRIEVE COORDINATES OF VERTEX AND HEIGHT VECTOR
C
      H(1)=ASTER(IH)
      H(2)=ASTER(IH+1)
      H(3)=ASTER(IH+2)
      V(1)=ASTER(IV)
      V(2)=ASTER(IV+1)
      V(3)=ASTER(IV+2)
C
C4    RETRIEVE RADIUS
C
      R=ASTER(IRR)
      RIN=-PINF
      ROUT=PINF
C
C5    COMPUTE R SQUARED
C
      RSQ=R*R
      LRO=0
      LRI=0
      TOP=0.
      POT=0.
C
C6    COMPUTE VECTOR DOT PRODUCTS
C
      HH=H(1)*H(1)+H(2)*H(2)+H(3)*H(3)
      VPH=H(1)*(V(1)-XB(1))+H(2)*(V(2)-XB(2))+H(3)*(V(3)-XB(3))
      WH=WB(1)*H(1)+WB(2)*H(2)+WB(3)*H(3)
C
C7    COMPUTE COEFFICIENT OF S SQUARED
C
      DEN=HH-WH*WH
      DO 10 I=1,3
      TOP=TOP+WB(I)*(XB(I)-V(I))
      POT=POT+(XB(I)-V(I))**2
   10 CONTINUE
      AMBD=-HH*TOP-WH*VPH
      UM=(POT-RSQ)*HH-VPH**2
      IF(WH)40,70,50
C
C8    SOLVE FOR RIN AND ROUT OF PLANE INTERSECTIONS
C
   40 CP=VPH/WH
```

FIG. 88. Source Listing, Subroutine RCC

```
      CM=(VPH+HH)/WH
      LCP=1
      LCM=2
      GOTO 60
   50 CP=(VPH+HH)/WH
      CM=VPH/WH
      LCM=1
      LCP=2
   60 IF(CP)300,80,80
   70 CP=PINF
      CM=-CP
      IF(VPH.GT.0.)GOTO 300
      IF(HH+VPH)300,90,90
   80 IF(ABS(DEN).GE.1.0E-6)GOTO 90
      R1=-PINF
      R2=PINF
      GOTO 100
   90 R1=0.
      R2=0.
      AMBDA=AMBD/DEN
      UMU=UM/DEN
      DISC=AMBDA**2-UMU
      IF(DISC.LE.0.)GOTO 300
      SD=SQRT(DISC)
C
C9    SOLVE FOR RIN AND ROUT OF QUADRATIC INTERSECTIONS
C
      R1=AMBDA-SD
      R2=AMBDA+SD
  100 IF(CM.GT.R1)GOTO 110
      RIN=R1
      LRI=3
      GOTO 120
  110 RIN=CM
      LRI=LCM
  120 IF(CP.LE.R2)GOTO 130
      ROUT=R2
      LRO=3
      GOTO 200
  130 ROUT=CP
      LRO=LCP
  200 IF(ABS(ROUT-RIN).LE.ROUT*1.0E-5)GOTO 300
      GOTO(210,210,220),LRO
C
C10   DETERMINE IF ROUT INTERSECTS PLANE WITHIN CYLINDER CROSS-SECTION
C
  210 F1=DEN*ROUT**2-2.*AMBD*ROUT+UM
      IF(F1)250,250,300
C
C11   DOES ROUT INTERSECT OF QUADRATIC SURFACE OCCUR BETWEEN PLANES
C
  220 F1=ROUT*WH-VPH
      IF(F1)300,250,230
  230 IF(F1.GT.HH) GOTO 300
  250 GOTO(260,260,270),LRI
C
C12   DETERMINE IF RIN INTERSECTS PLANE WITHIN CYLINDER CROSS-SECTION
C
  260 F1=DEN*RIN**2-2.*AMBD*RIN+UM
      IF(F1)310,310,300
```

FIG. 88. (Contd.)

```
C
C13    DOES RIN INTERSECT OF QUADRATIC SURFACE OCCUR BETWEEN PLANES
C
  270 F1=RIN*WH-VPH
      IF(F1)300,310,280
  280 IF(F1.LE.HH)GOTO 310
C
C14    RAY MISSES BODY
C
  300 RIN=PINF
      ROUT=-PINF
      LRO=0
      LRI=0
  310 RETURN
      END
C
C
```

FIG. 88. (Concluded)

```
      SUBROUTINE REC
      DIMENSION V(3),H(3),A(3),B(3)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMCN/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
C
C1    RETRIEVE LOCATION OF REC VEXTEX AND HEIGTH VECTOR COORDINATES
C
      CALL UN2(LOCDA,IV,IH)
      LOC=LOCDA+1
C
C2    RETRIEVE LOCATION OF REC COORDINATES FOR AXES
C
      CALL UN2(LOC,IA,IB)
C
C3    RETRIEVE COORDINATES OF VERTEX, HEIGHT VECTOR, SEMI-MAJOR AXIS
C     AND SEMI-MINOR AXIS
C
      V(1)=ASTER(IV)
      V(2)=ASTER(IV+1)
      V(3)=ASTER(IV+2)
      H(1)=ASTER(IH)
      H(2)=ASTER(IH+1)
      H(3)=ASTER(IH+2)
      A(1)=ASTER(IA)
      A(2)=ASTER(IA+1)
      A(3)=ASTER(IA+2)
      B(1)=ASTER(IB)
      B(2)=ASTER(IB+1)
      B(3)=ASTER(IB+2)
      RIN=-PINF
      ROUT=PINF
      LRO=0
      LRI=0
C
C4    COMPUTE DOT PRODUCTS OF A.A AND B.B
C
      AA=A(1)*A(1)+A(2)*A(2)+A(3)*A(3)
      BB=B(1)*B(1)+B(2)*B(2)+B(3)*B(3)
C
C5    COMPUTE (V-XB) FOR X,Y,Z COORDINATES
C
      V1XB1=V(1)-XB(1)
      V2XB2=V(2)-XB(2)
      V3XB3=V(3)-XB(3)
C
C6    TRANSFORM XB(X,Y,Z) TO THE COORDINATES OF THE REC
C
      VPA=V1XB1*A(1)+V2XB2*A(2)+V3XB3*A(3)
      VPB=V1XB1*B(1)+V2XB2*B(2)+V3XB3*B(3)
C
C7    TRANSFORM WB(X,Y,Z) TO THE COORDINATES OF THE REC
C
      WBA=WB(1)*A(1)+WB(2)*A(2)+WB(3)*A(3)
      WBB=WB(1)*B(1)+WB(2)*B(2)+WB(3)*B(3)
      WBAWBA=WBA*WBA
      WBBWBB=WBB*WBB
      AAAA=AA*AA
```

FIG. 89. Source Listing, Subroutine REC

```
      BBBB=BB*BB
      AMBD=WBA*VPA*BBBB+WBB*VPB*AAAA
      UM=BBBB*VPA*VPA+AAAA*VPB*VPB-AAAA*BBBB
      DEN=WBA*WBA*BBBB+WBB*WBB*AAAA
      IF(ABS(DEN).LE.1.0E-6)GOTO 10
      AMBDA=AMBD/DEN
      UMU=UM/DEN
      DISC=AMBDA**2-UMU
      IF(DISC.LE.0.)GOTO 300
C
C8    COMPUTE THE INTERSECT POINTS ON THE QUADRATIC SURFACE
C
      SD=SQRT(DISC)
      R1=AMBDA-SD
      R2=AMBDA+SD
      GOTO 20
   10 R1=-PINF
      R2=PINF
   20 HH=H(1)*H(1)+H(2)*H(2)+H(3)*H(3)
      WH=WB(1)*H(1)+WB(2)*H(2)+WB(3)*H(3)
      VPH=V1XB1*H(1)+V2XB2*H(2)+V3XB3*H(3)
C
C9    DETERMINE IF RAY PARALLEL TO PLANAR SURFACES
C
      IF(WH)40,70,50
   40 IF(VPH.GE.0.)GOTO 300
C
C10   COMPUTE THE INTERSECT POINTS ON THE PLANAR SURFACES
C
      CP=VPH/WH
      CM=(VPH+HH)/WH
      LCP=1
      LCM=2
      GOTO 100
   50 VPHHH=VPH+HH
      IF(VPHHH.LE.0.)GOTO 300
      CP=VPHHH/WH
      CM=VPH/WH
      LCM=1
      LCP=2
      GOTO 100
   70 CP=PINF
      CM=-CP
  100 IF(CM.GT.R1)GOTO 110
C
C11   RIN FOR THE QUADRATIC SURFACE
C
      RIN=R1
      LRI=3
      GOTO 120
C
C12   RIN FOR A PLANAR SURFACE
C
  110 RIN=CM
      LRI=LCM
  120 IF(CP.LE.R2)GOTO 130
C
C13   ROUT FOR THE QUADRATIC SURFACE
C
      ROUT=R2
```

FIG. 89. (contd.)

```
      LRO=3
      GOTO 200
C
C14   ROUT FOR A PLANAR SURFACE
C
  130 ROUT=CP
      LRO=LCP
  200 IF(ABS(ROUT-RIN).LE.ROUT*1.0E-5)GOTO 300
      GOTO(210,210,220),LRO
C
C15   DETERMINE IF ROUT OF PLANAR SURFACE OCCURS WITHIN ELLIPTIC
C     CROSS-SECTION
C
  210 F1=DEN*ROUT**2-2.*AMBD*ROUT+UM
      IF(F1)250,250,300
C
C16   DETERMINE IF ROUT OF QUADRATIC OCCURS BETWEEN PLANAR SURFACES
C
  220 F1=ROUT*WH-VPH
      IF(F1)300,250,230
  230 IF(F1.GT.HH)GOTO 300
  250 GOTO(260,260,270),LRI
C
C17   DETERMINE IF RIN OF PLANE WITHIN ELLIPTIC CROSS SECTION
C
  260 F1=DEN*RIN**2-2.*AMBD*RIN+UM
      IF(F1)310,310,300
C
C18   DETERMINE IF RIN OF QUADRATIC SURFACE BETWEEN PLANAR SURFACES
C
  270 F1=RIN*WH-VPH
      IF(F1)300,310,280
  280 IF(F1.LE.HH)GOTO 310
C
C19   RAY MISSES BODY
C
  300 RIN=PINF
      ROUT=-PINF
      LRI=0
      LRO=0
  310 RETURN
      END
C
C
```

FIG. 89. (Concluded)

```
      SUBROUTINE TRC
      DIMENSION V(3),H(3)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1 LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      EQUIVALENCE(MASTER,ASTER)
C
C1    RETRIEVE LOCATION OF TRC VERTEX AND HEIGHT VECTOR COORDINATES
C
      CALL UN2(LOCDA,IV,IH)
      LOC=LOCDA+1
C
C2    RETRIEVE LOCATION OF TRC RADII FOR LOWER BASE AND UPPER BASE
C
      CALL UN2(LOC,IRB,IRTOP)
C
C3    RETRIEVE COORDINATES OF VERTEX AND HEIGHT VECTOR
C
      V(1)=ASTER(IV)
      V(2)=ASTER(IV+1)
      V(3)=ASTER(IV+2)
      H(1)=ASTER(IH)
      H(2)=ASTER(IH+1)
      H(3)=ASTER(IH+2)
C
C4    RETRIEVE RADII OF LOWER AND UPPER BASES
C
      RB=ASTER(IRB)
      RT=ASTER(IRTOP)
      RIN=-PINF
      ROUT=PINF
      LRO=0
      LRI=0
      INTSEC=0
      INTR1=0
      INTR2=0
C
C5    COMPUTE COORDINATES OF (V-XB)
C
      V1XB1=V(1)-XB(1)
      V2XB2=V(2)-XB(2)
      V3XB3=V(3)-XB(3)
C
C6    COMPUTE DOT PRODUCTS
C
      PVPV=V1XB1*V1XB1+V2XB2*V2XB2+V3XB3*V3XB3
      VPW=V1XB1*WB(1)+V2XB2*WB(2)+V3XB3*WB(3)
      WH =WB(1)*H(1)+WB(2)*H(2)+WB(3)*H(3)
      VPH=V1XB1*H(1)+V2XB2*H(2)+V3XB3*H(3)
      HH=H(1)*H(1)+H(2)*H(2)+H(3)*H(3)
      RTRB=RT-RB
C
C7    COMPUTE C2 QUANTITY OF QUADRATIC EQUATION
C
      RBRTVP=RB-VPH*RTRB/HH
      VPHHH=VPH+HH
      UM=HH*(PVPV-RBRTVP**2)-VPH*VPH
      AMBD=HH*VPW-WH*(VPH-RTRB*RBRTVP)
```

FIG. 90. Source Listing, Subroutine TRC

```
      DEN=HH-WH**2*(1.+RTRB**2/HH)
C
C8    TEST FOR RAY PARALLEL TO EITHER SIDE OF CONE
C
      IF(ABS(DEN).GT.1.0E-6)GOTO 40
      IF(RTRB.EQ.0.)GOTO 200
C
C9    COMPUTE INTERSECT WITH QUADRATIC SURFACE FOR RAY PARALLEL TO SIDE
C
      R2=UM/(2.*AMBD)
      F1=R2*WH-VPH
C
C10   TEST IF INTERSECT BETWEEN PLANAR SURFACES
C
      IF(F1.LT.0.)GOTO 200
      IF(F1.GT.HH)GOTO 200
      INTSEC=INTSEC+1
      IF(WH.LE.0.)GOTO 10
      IF(RTRB)20,20,30
   10 IF(RTRB)30,30,20
C
C11   ASSIGN SURFACE EXIT NUMBER AND ROUT FOR QUADRATIC SURFACE
C
   20 LRO=3
      ROUT=R2
      GOTO 250
C
C12   ASSIGN SURFACE ENTRY NUMBER AND RIN FOR QUADRATIC SURFACE
C
   30 LRI=3
      RIN=R2
      INTSEC=INTSEC+1
      GOTO 210
C
   40 AMBDA=AMBD/DEN
      UMU=UM/DEN
      DISC=AMBDA**2-UMU
      IF(DISC)350,200,50
C
C13   SOLVE FOR VALUES OF QUADRATIC EQUATION
C
   50 SD=SQRT(DISC)
      R1=AMBDA-SD
      R2=AMBDA+SD
      F1=R2*WH-VPH
C
C14   TEST FOR INTERSECT BETWEEN PLANAR SURFACES
C
      IF(F1.LT.0.)GOTO 60
      IF(F1.LE.HH)INTR2=INTR2+1
   60 F1=R1*WH-VPH
      IF(F1.LT.0.)GOTO 70
      IF(F1.LE.HH)GOTO 80
   70 IF(INTR2.LT.1)GOTO 200
      ROUT=R2
      RIN=R2
      LRO=3
      LRI=3
      INTSEC=INTSEC+1
      GOTO 200
```

FIG. 90. (Contd.)

```
   80 INTR1=INTR1+1
      IF(INTR2.GE.1)GOTO 90
      ROUT=R1
      RIN=R1
      LRO=3
      LRI=3
      INTSEC=INTSEC+1
      GOTO 200
   90 IF(R1-R2)100,350,110
C
C15    COMPUTE RIN AND ROUT FOR QUADRATIC SURFACE
C
  100 RIN=R1
      ROUT=R2
      LRO=3
      LRI=3
      GOTO 300
  110 RIN=R2
      ROUT=R1
      LRO=3
      LRI=3
      GOTO 300
C
  200 IF(WH)210,350,250
  210 IF(VPH.GE.0.)GOTO 350
      CP=VPH/WH
      F1=CP*CP-2.*CP*VPW+PVPV-RB*RB
      IF(F1.GT.0.)GOTO 220
C
C16    COMPUTE ROUT FOR EXIT FROM V-PLANE SURFACE
C
      INTSEC=INTSEC+1
      ROUT=CP
      LRO=1
      IF(INTSEC.GE.2)GOTO 300
  220 CM=VPHHH/WH
      F1=CM*CM-2.*((VPW+WH)*CM-VPH)+HH*PVPV-RT*RT
      IF(F1.GT.0.)GOTO 350
C
C17    COMPUTE RIN FOR ENTRY INTO V+H PLANE SURFACE
C
      RIN=CM
      LRI=2
      GOTO 300
  250 IF(VPHHH.LT.0.)GOTO 350
      CP=VPHHH/WH
      F1=CP*CP-2.*((VPW+WH)*CP-VPH)+HH*PVPV-RT*RT
      IF(F1.GT.0.)GOTO 260
C
C18    COMPUTE ROUT FOR EXIT FROM V+H PLANE SURFACE
C
      INTSEC=INTSEC+1
      ROUT=CP
      LRO=2
  260 IF(INTSEC.GE.2)GOTO 300
      CM=VPH/WH
      F1=CM*CM-2.*CM*VPW+PVPV-RB*RB
      IF(F1.GT.0.)GOTO 350
```

FIG. 90. (Cont.)

```
C
C19    COMPUTE RIN FOR ENTRY INTO V-PLANE SURFACE
C
       RIN=CM
       LRI=1
C
  300  IF(ABS(ROUT-RIN)-ROUT*1.0E-5)350,350,360
C
C20    RAY MISSES TRC
C
  350  RIN=PINF
       ROUT=-PINF
       LRI=0
       LRO=0
  360  RETURN
       END
C
C
```

FIG. 90. (Concluded)

```
      SUBROUTINE ELL
      DIMENSION FOCIA(3),FOCIB(3)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGO,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      EQUIVALENCE (ASTER,MASTER)
C
C1    RETRIEVE LOCATION OF ELLIPSE FOCI AND LENGTH STORAGE POSITIONS
C
      CALL UN2(LOCDA,IV1,IV2)
      IRR=MASTER(LOCDA+1)
      FOCIA(1)=ASTER(IV1)
      FOCIA(2)=ASTER(IV1+1)
      FOCIA(3)=ASTER(IV1+2)
      FOCIB(1)=ASTER(IV2)
      FOCIB(2)=ASTER(IV2+1)
      FOCIB(3)=ASTER(IV2+2)
      C=ASTER(IRR)
      RIN=PINF
      ROUT=-PINF
C
C2    COMPUTE COORDINATES FOR VECTOR D1 AND D2
C
      D1X=XB(1)-FOCIA(1)
      D1Y=XB(2)-FOCIA(2)
      D1Z=XB(3)-FOCIA(3)
      D2X=XB(1)-FOCIB(1)
      D2Y=XB(2)-FOCIB(2)
      D2Z=XB(3)-FOCIB(3)
C
C3    COMPUTE DOT PRODUCTS
C
      A1=2.*(D1X*WB(1)+D1Y*WB(2)+D1Z*WB(3))
      A2=2.*(D2X*WB(1)+D2Y*WB(2)+D2Z*WB(3))
      B1=D1X*D1X+D1Y*D1Y+D1Z*D1Z
      B2=D2X*D2X+D2Y*D2Y+D2Z*D2Z
C
C4    COMPUTE A AND B
C
      AA=(A2-A1)/(2.*C)
      BB=(C*C+B2-B1)/(2.*C)
C
C5    COMPUTE LAMBDA AND MU
C
      ALAMD=AA*AA-1.
      ALAM1=(AA*BB-.5*A2)/ALAMD
      U=(BB*BB-B2)/ALAMD
C
C6    COMPUTE RIN AND ROUT
C
      DISCRM=ALAM1*ALAM1-U
      IF(DISCRM.LE.0.)RETURN
      SQRTDI=SQRT(DISCRM)
      RIN=-ALAM1-SQRTDI
      ROUT=-ALAM1+SQRTDI
      RETURN
      END
C
C
```

FIG. 91. Source Listing, Subroutine ELL

```
      SUBROUTINE RAW
      DIMENSION H1(3),H2(3),H3(3),V(3),ASQ(3),PV(4),G(3)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
C
C1    RETRIEVE LOCATIONS OF VERTEX AND LENGTH VECTORS
C
      CALL UN2(LOCDA,IV,IH1)
      LOC=LOCDA+1
      CALL UN2(LOC,IH2,IH3)
      H1(1)=ASTER(IH1)
      H1(2)=ASTER(IH1+1)
      H1(3)=ASTER(IH1+2)
      H2(1)=ASTER(IH2)
      H2(2)=ASTER(IH2+1)
      H2(3)=ASTER(IH2+2)
      H3(1)=ASTER(IH3)
      H3(2)=ASTER(IH3+1)
      H3(3)=ASTER(IH3+2)
      V(1)=ASTER(IV)
      V(2)=ASTER(IV+1)
      V(3)=ASTER(IV+2)
      RIN=-PINF
      ROUT=PINF
      CM=-PINF
      CP=PINF
      L=0
      L1=0
      K=0
      LRI=0
      LRO=0
C
C2    COMPUTE A
C
      ASQ(1)=H1(1)*H1(1)+H1(2)*H1(2)+H1(3)*H1(3)
      ASQ(2)=H2(1)*H2(1)+H2(2)*H2(2)+H2(3)*H2(3)
      ASQ(3)=H3(1)*H3(1)+H3(2)*H3(2)+H3(3)*H3(3)
C
C3    COMPUTE P
C
      XB1V1=XB(1)-V(1)
      XB2V2=XB(2)-V(2)
      XB3V3=XB(3)-V(3)
      PV(1)=XB1V1*H1(1)+XB2V2*H1(2)+XB3V3*H1(3)
      PV(2)=XB1V1*H2(1)+XB2V2*H2(2)+XB3V3*H2(3)
      PV(3)=XB1V1*H3(1)+XB2V2*H3(2)+XB3V3*H3(3)
C
C4    COMPUTE G
C
      G(1)=WB(1)*H1(1)+WB(2)*H1(2)+WB(3)*H1(3)
      G(2)=WB(1)*H2(1)+WB(2)*H2(2)+WB(3)*H2(3)
      G(3)=WB(1)*H3(1)+WB(2)*H3(2)+WB(3)*H3(3)
C
      DO 140 I=1,2
      IF(G(I))10,110,60
   10 IF(-PV(I))20,400,400
```

FIG. 92. Source Listing, Subroutine RAW

```
C
C5     COMPUTE S1 OR S3
C
   20 TEMP=-PV(I)/G(I)
      IF(TEMP-CP)30,130,130
   30 CP=TEMP
      L=I
      GOTO(40,50),I
   40 LRO=3
      GOTO 130
   50 LRO=1
      GOTO 130
   60 IF(-PV(I).LE.0.)GOTO 130
C
C6     COMPUTE S1 OR S3
C
      TEMP=-PV(I)/G(I)
      IF(TEMP.LE.CM)GOTO 130
      CM=TEMP
      K=I
      GOTO(90,100),I
   90 LRI=3
      GOTO 130
  100 LRI=1
      GOTO 130
C
  110 IF(PV(I).LE.0.)GOTO 810
      IF(PV(I).GE.ASQ(I))GOTO 810
  130 L1=L1+I
  140 CONTINUE
C
      IF(G(3))150,210,230
C
C7     COMPUTE S6
C
  150 TEMP=ASQ(3)-PV(3)
      IF(TEMP.GE.0.)GOTO 180
      TEMP=TEMP/G(3)
      IF(TEMP.LE.CM)GOTO 190
      CM=TEMP
      K=3
      LRI=6
  180 IF(-PV(3))190,400,400
C
C8     COMPUTE S5
C
  190 TEMP=-PV(3)/G(3)
      IF(TEMP.GE.CP)GOTO 290
      CP=TEMP
      L=3
      LRO=5
      GOTO 290
C
  210 IF(PV(3).LE.0.)GOTO 400
      IF(PV(3)-ASQ(3))290,290,400
C
C9     COMPUTE S5
C
  230 IF(-PV(3).LE.0.)GOTO 260
      TEMP=-PV(3)/G(3)
```

FIG. 92. (Contd.)

```
      IF(TEMP.LE.CM)GOTO 260
      CM=TEMP
      K=3
      LRI=5
C
C10   COMPUTE S6
C
  260 TEMP=ASQ(3)-PV(3)
      IF(TEMP.LE.0.)GOTO 400
      TEMP=TEMP/G(3)
      IF(TEMP.GE.CP)GOTO 290
      CP=TEMP
      L=3
      LRO=6
C
C11   COMPUTE S2
C
  290 AG=ASQ(2)*G(1)+ASQ(1)*G(2)
      PV(4)=PV(1)*ASQ(2)+PV(2)*ASQ(1)
      TOP=ASQ(1)*ASQ(2)-PV(4)
      IF(AG)310,350,330
  310 TEMP=TOP/AG
      IF(TEMP.LE.CM)GOTO 380
      CM=TEMP
      K=4
      LRI=2
      GOTO 380
C
  330 IF(TOP.LT.0.)GOTO 400
      TEMP=TOP/AG
      IF(TEMP-CP)370,380,380
C
  350 IF(PV(4).LE.0.)GOTO 400
      IF(-TOP)380,400,400
  370 CP=TEMP
      L=4
      LRO=2
  380 IF(L*K.LE.0)GOTO 400
      ROUT=CP
      RIN=CM
C
  400 IF(ROUT.GE.PINF)GOTO 810
      IF(ROUT.LE.0.)GOTO 810
      IF(RIN.GE.ROUT)GOTO 810
      IF(ABS(RIN-ROUT).GT.ROUT*1.0E-5)GOTO 820
C
  810 ROUT=-PINF
      RIN=PINF
      LRO=0
      LRI=0
  820 RETURN
      END
C
C
```

FIG. 92. (Concluded)

```
      SUBROUTINE ARB
      DIMENSION AA(6,4),XP(3)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
C
C1    RETRIEVE PLANAR EQUATIONS FROM ASTER ARRAY
C
      LOC=LOCDA-1
      DO 10 I=1,6
      LOC=LOC+1
      CALL UN2(LOC,LD,LC)
      AA(I,1)=ASTER(LC)
      AA(I,2)=ASTER(LC+1)
      AA(I,3)=ASTER(LC+2)
      AA(I,4)=ASTER(LD)
   10 CONTINUE
      RIN=-PINF
      ROUT=PINF
      LRO=0
      LRI=0
      S1=0.
      S2=0.
      L1=0
      L2=0
      DO 70 I=1,6
C
C2    COMPUTE NUMERATOR AND DENOMINATOR OF DISTANCE EQUATION
C
      D=AA(I,4)
      SNUM=-D-AA(I,1)*XB(1)-AA(I,2)*XB(2)-AA(I,3)*XB(3)
      SDEN=AA(I,1)*WB(1)+AA(I,2)*WB(2)+ AA(I,3)*WB(3)
      IF(SDEN)20,70,30
   20 IF(SNUM)40,70,70
   30 IF(SNUM)70,70,40
C
C3    COMPUTE INTERSECT DISTANCE
C
   40 S=SNUM/SDEN
      DO 50 K=1,3
      XP(K)=XB(K)+S*WB(K)
   50 CONTINUE
C
C4    TEST IF INTERSECT POINT IS ON ARB
C
      DO 60 J=1,6
      IF(I.EQ.J)GOTO 60
      T=AA(J,1)*XP(1)+AA(J,2)*XP(2)+AA(J,3)*XP(3)+AA(J,4)
      IF(ABS(T).LE.1.0E-6)T=0.
      IF(T.LT.0.)GOTO 70
   60 CONTINUE
      IF(L1.GT.0)GOTO 65
      L1=I
      S1=S
      GOTO 70
   65 IF(ABS(S1-S).GT.1.0E-6)GOTO 100
   70 CONTINUE
C
```

FIG. 93. Source Listing, Subroutine ARB

```
      IF(L1)200,200,150
  100 S2=S
      L2=I
      IF(ABS(S1-S2).LE.S1*1.0E-5)GOTO 200
      IF(S1-S2)110,200,120
  110 RIN=S1
      ROUT=S2
      LRI=L1
      LRO=L2
      RETURN
  120 RIN=S2
      LRI=L2
  130 ROUT=S1
      LRO=L1
      RETURN
  150 DO 160 J=1,6
      IF(L1.EQ.J)GOTO 160
      T1=AA(J,1)*XB(1)+AA(J,2)*XB(2)+AA(J,3)*XB(3)+AA(J,4)
      IF(ABS(T1).LE.1.0E-6)T1=0.
      IF(T1.LT.0.)GOTO 200
  160 CONTINUE
      GOTO 130
C
  200 RIN=PINF
      ROUT=-PINF
      LRI=0
      LRO=0
      RETURN
      END
C
C
```

FIG. 93. (Concluded)

```
      SUBROUTINE TEC
      DIMENSION VXB(3),H(3),HN(3),AA(3),BB(3)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGO,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      EQUIVALENCE(ASTER,MASTER)
C
C1    RETRIEVE LOCATION OF VERTEX AND HEIGHT VECTOR COORDINATES
C
      CALL UN2(LOCDA,IV,IH)
      LOC=LOCDA+1
C
C2    RETRIEVE LOCATION OF NORMAL AND AXES COORDINATES
C
      CALL UN2(LOC,IN,IA)
      LOC=LOC+1
C
C3    RETRIEVE LOCATION OF LENGTHS OF SEMI-MAJOR AXIS AND
C     SEMI-MINOR AXIS OF BASE ELLIPSE
C
      CALL UN2(LOC,IR1,IR2)
C
C4    RETRIEVE LOCATION OF THE RATIO OF THE LARGER TO SMALLER ELLIPSE
C
      IR3=MASTER(LOC+1)
C
C5    RETRIEVE COORDINATES OF VERTEX AND COMPUTE COORDINATES
C     OF (V-XB) VECTOR
C
      VXB(1)=ASTER(IV)-XB(1)
      VXB(2)=ASTER(IV+1)-XB(2)
      VXB(3)=ASTER(IV+2)-XB(3)
C
C6    RETRIEVE COORDINATES OF HEIGHT VECTOR
C
      H(1)=ASTER(IH)
      H(2)=ASTER(IH+1)
      H(3)=ASTER(IH+2)
C
C7    RETRIEVE COORDINATES OF NORMAL TO BASE ELLIPSE
C
      HN(1)=ASTER(IN)
      HN(2)=ASTER(IN+1)
      HN(3)=ASTER(IN+2)
C
C8    RETRIEVE COORDINATES OF SEMI-MAJOR AXIS OF BASE ELLIPSE
C
      AA(1)=ASTER(IA)
      AA(2)=ASTER(IA+1)
      AA(3)=ASTER(IA+2)
C
C9    COMPUTE SEMI-MINOR AXIS UNIT VECTOR OF BASE ELLIPSE
C
      CALL CROSS(BB,AA,HN)
```

FIG. 94. Source Listing, Subroutine TEC

```
C
C10    RETRIEVE LENGTHS OF SEMI-MAJOR AND SEMI-MINOR AXES OF BASE
C      ELLIPSE AND RATIO OF LARGER TO SMALLER ELLIPSE
C
       R1=ASTER(IR1)
       R2=ASTER(IR2)
       RR=ASTER(IR3)
C
C11    COMPUTE LENGTHS OF SEMI-MAJOR AND SEMI-MINOR AXES OF TOP ELLIPSE
C
       R3=R1/RR
       R4=R2/RR
C
C12    START OF COMPUTATIONS FOR DOT PRODUCTS
C
       HDN=DOT(H,HN)
       HDA=DOT(H,AA)
       HDB=DOT(H,BB)
       WDN=DOT(WB,HN)
       WDA=DOT(WB,AA)
       WDB=DOT(WB,BB)
       VXBDN=DOT(VXB,HN)
       VXBDA=DOT(VXB,AA)
       VXBDB=DOT(VXB,BB)
C
C13    TEST TO DETERMINE IF RAY IS PARALLEL TO TOP AND BASE PLANES
C
       IF(ABS(WDN).GT.0.0001)GOTO 20
C
C14    COMPUTE RATIO ON NORMAL TO HEIGHT OF HIT
C
       GAMMA = -VXBDN/HDN
       IF(GAMMA.LT.0.0  .OR.  GAMMA.GT.1.0) GOTO 500
       A=GAMMA*R3+R1*(1.-GAMMA)
       B=GAMMA*R4+R2*(1.-GAMMA)
       ASQ=A*A
       BSQ=B*B
       TA=VXBDA+GAMMA*HDA
       TB=VXBDB+GAMMA*HDB
       DEN=BSQ*WDA*WDA+ASQ*WDB*WDB
       IF(ABS(DEN).LE.0.0001)GOTO 500
       AMBDA=BSQ*WDA*TA+ASQ*WDB*TB
       UM=BSQ*TA*TA+ASQ*TB*TB-ASQ*BSQ
       DISC=AMBDA*AMBDA-DEN*UM
       IF(DISC.LT.0.0) GOTO 500
       DISC=SQRT(DISC)
C
C15    COMPUTE RIN AND ROUT AND ASSIGN SURFACE NUMBER FOR
C      RIN AND ROUT WITH QUADRATIC SURFACE
C
       RIN=(AMBDA-DISC)/DEN
       ROUT=(AMBDA+DISC)/DEN
       LRI=3
       LRO=3
       GOTO 400
C
C16    SOLVE FOR TERMS IN QUADRATIC EQUATION
C
   20  TAU=(R1/R2)**2
       R2SQ=R2*R2
```

FIG. 94. (Contd.)

```
      TR2SQ=TAU*R2SQ
      TR4R2=TAU*(R2-R4)**2
      TRR4R2=TR2SQ-TAU*R2*R4
      BETA=VXBDN/WDN
      ALPHA=HDN/WDN
      TA1=ALPHA*WDA-HDA
      TB1=ALPHA*WDB-HDB
      TA2=VXBDA-BETA*WDA
      TB2=VXBDB-BETA*WDB
      DEN  =TA1*TA1+TAU*TB1*TB1-TR4R2
      AMBDA=TA1*TA2+TAU*TB1*TB2-TRR4R2
      UM   =TA2*TA2+TAU*TB2*TB2-TR2SQ
      IF(ABS(DEN).GT.0.0001)GOTO 150
      IF(R1.EQ.R3)GOTO 100
      IF(AMBDA.NE.0.0)GOTO 110
C
C17   THE RAY MISSES THE QUADRATIC SURFACE OF THE TEC
C
  100 S1=-PINF
      S2=PINF
      GOTO 200
  110 T=UM/(2.*AMBDA)
C
C18   COMPUTE DISTANCE TO INTERSECT WITH QUADRATIC SURFACE
C
  120 S=BETA+ALPHA*T
      F=S*WDN-VXBDN
      IF(ABS(F).LE.0.0001)GOTO 125
      IF(F.LT.0.0)GOTO 100
      IF(ABS(F-HDN).LE.0.0001)GOTO 125
      IF(F.GT.HDN)GOTO 100
  125 IF(WDN)130,500,140
C
C19   ASSIGN TEMPORARY VALUES TO RIN AND ROUT PER DIRECTION OF RAY
C
  130 S1=S
      S2=PINF
      GOTO 200
  140 S1=-PINF
      S2=S
      GOTO 200
C
C20   RAY PARALLEL TO SIDE
C
  150 DISC=AMBDA*AMBDA-DEN*UM
      IF(ABS(DISC).GT.0.0001)GOTO 155
      T=AMBDA/DEN
      GOTO 120
  155 IF(DISC.LT.0.0)GOTO 500
C
C21   SOLVE FOR TWO INTERSECTS WITH QUADRATIC SURFACE
C
      DISC=SQRT(DISC)
      T1=(AMBDA-DISC)/DEN
      T2=(AMBDA+DISC)/DEN
      S1=BETA+ALPHA*T1
      S2=BETA+ALPHA*T2
      IF(WDN.GE.0.0)GOTO 160
      T=S1
      S1=S2
      S2=T
```

FIG. 94. (Contd.)

```
C
C22   DETERMINE IF SIDE INTERSECTION BETWEEN PLANES
C
  160 F=S1*WDN-VXBDN
      IF(F.LT.0.0)GOTO 170
      IF(F.LE.HDN)GOTO 180
  170 S1=-PINF
  180 F=S2*WDN-VXBDN
      IF(F.LT.0.0)GOTO 190
      IF(F.LE.HDN)GOTO 200
  190 S2=PINF
C
  200 IF(WDN) 220,210,230
C
C23   RAY PARALLEL TO PLANES
C
  210 SI=-PINF
      SO=PINF
      GOTO 300
C
C24   COMPUTE INTERSECTIONS WITH PLANE SURFACES
C
  220 SI=BETA+ALPHA
      SO=BETA
      LI=2
      LO=1
      GOTO 240
C
  230 SI=BETA
      SO=BETA+ALPHA
      LI=1
      LO=2
  240 IF(SO.LT.0.0) GOTO 500
C
C25   DETERMINE WHICH SURFACE IS HIT
C
  300 IF(SI.GE.S1)GOTO 310
      IF(ABS(SI-S1).LE.0.0001)GOTO 310
      RIN=S1
      LRI=3
      GOTO 350
  310 RIN=SI
      LRI=LI
  350 IF(SO.LE.S2) GOTO 360
      IF(ABS(SO-S2).LE.0.0001)GOTO 360
      ROUT=S2
      LRO=3
      GOTO 400
  360 ROUT=SO
      LRO=LO
C
  400 IF(RIN.GE.ROUT)GOTO 500
      IF(ABS(RIN-ROUT).LE.0.0001)GOTO 500
      IF(ROUT.LE.0.0)GOTO 500
C
      S=ROUT
      I=1
      GOTO(420,430,410),LRO
  410 S=RIN
      I=2
      GOTO(420,430,480),LRI
```

FIG. 94. (Contd.)

```
C
C26   DETERMINE IF INTERSECTION WITH BASE PLANE LIES WITHIN
C     CROSS SECTION OF BASE ELLIPSE
C
  420 F1=S*WDA-VXBDA
      F2=S*WDB-VXBDB
      F=F1*F1/(R1*R1)+F2*F2/(R2*R2)
      IF(F.GT.1.0001)GOTO 500
      GOTO(410,480),I
C
  430 IF(R3.EQ.0.0.OR.R4.EQ.0.0)GOTO 480
C
C27   DETERMINE IF INTERSECTION WITH TOP PLANE LIES WITHIN
C     CROSS SECTION OF TOP ELLIPSE
C
      F1=S*WDA-VXBDA-HDA
      F2=S*WDB-VXBDB-HDB
      F=F1*F1/(R3*R3)+F2*F2/(R4*R4)
      IF(F.GT.1.0001)GOTO 500
      GOTO(410,480),I
C
C28   RAY ORIGINATES WITHIN TEC
C
  480 IF(RIN.GT.0.0001)RETURN
      RIN=-.0001
      LRI=0
      RETURN
C
C29   RAY MISSES TEC
C
  500 RIN=PINF
      ROUT=-PINF
      LRI=0
      LRO=0
      RETURN
      END
C
C
```

FIG. 94. (Concluded)

```
      SUBROUTINE ARS
C
C          SURROUTINE COMPUTES INTERSECTIONS OF RAY WITH ARBITRARY
C          SURFACE - ARS
C
      DIMENSION W(3),UW(3),VW(3),WXB(3),WN(3),
     1     HIT(20),ORMAL(3,20),ISURF(20)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1     LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/DAVIS/IGRID,LOOP,INORM
      COMMON/WHICH/NBO
      EQUIVALENCE (MASTER,ASTER)
C
  901 FORMAT(1H0,12HERROR IN ARS,I5,4X,22HNUMBER OF HITS .GT. 20)
  902 FORMAT(1H0,12HERROR IN ARS,I5,4X,21HNUMBER OF HITS IS ODD,
     1          2X,6H(NHIT=,I5,1H) )
  903 FORMAT(1H0,12HERROR IN ARS,I5,4X,27HWRONG SEQUENCE IN HIT TABLE,
     1          2X,6H(NHIT=,I5,1H) )
  910 FORMAT(5X,46HTHIS ERROR USUALLY MEANS THE ARS IS NOT CLOSED /
     1       9X,3HHIT,5X,7HSURFACE / (F12.4,I12) )
C
C          ARS DATA STORAGE IN ASTER ARRAY -
C
C     LOCARS
C       +0      NP   - NUMBER OF POINTS
C       +1      NHIT - NUMBER OF HITS
C       +2      )
C        .      )    - RESERVE 80 WORDS FOR HITS (4 PER HIT)
C        .      )    - ALLOWS FOR 10 PAIRS OF RIN/ROUT
C      +81      )
C          LOCHTS = LOCARS + 2
C            +0      S     - DISTANCE FROM START POINT XB TO TRIANGLE HIT
C            +1      NX    - DIRECTION COSINES OF NORMAL TO TRIANGLE HIT
C            +2      NY    -    (NX,NY,NZ)
C            +3      NZ    -
C      +82      X    - 4 WORDS PER POINT (X,Y,Z)
C        .      Y    - NP IS TOTAL NUMBER OF POINTS
C        .      Z    -
C        .      FLAG - SET = -1 TO SIGNAL LINE OR POINT TRIANGLE
C          LOC   = LOCARS + 82
C           +0     X
C           +1     Y
C           +2     Z
C           +3     FLAG
C
C          LOC   = LOC   + 4 FOR NEXT TRIANGLE
C
C
C          DETERMINE IF RE ENTRY
C
      LOCARS=MASTER(LOCDA)
      LOCHTS=LOCARS+2
      IF(LOOP.NE.KLOOP)GOTO 100
```

FIG. 95. Source Listing, Subroutine ARS

```
C
C          REENTRY
C
      NHIT=MASTER(LOCARS+1)
      IF(NHIT.LE.0)RETURN
      LOC=LOCHTS
C
C          MOVE INTERSECT DATA TO HIT ARRAY
C
      DO 10 I=1,NHIT
      HIT(I)=ASTER(LOC)
      LOC=LOC+4
   10 CONTINUE
      GOTO 600
C
C          NOT A REENTRY - ZERO INTERSECT DATA SECTION OF ASTER ARRAY
C
  100 NHIT=0
      N=1
      IF(NASC.EQ.-2)GOTO 400
      L1=LOCHTS
      L2=LOCHTS+79
      DO 110 L=L1,L2
      ASTER(L)=0.
  110 CONTINUE
C
C          COMPUTE FOR HIT ON TRIANGLE IT,  STORE FLAG AT ASTER(LOC+3)
C              -1 TRIANGLE IS A POINT OR LINE (DE-GENERATE)
C               0 NON-DEGENERATE, COMPUTE INTERSECT DATA
C
  400 LOC=LOCARS+82
      NT=MASTER(LOCARS)-2
      DO 499 IT=1,NT
      IF(ASTER(LOC+3).LT.0.0)GOTO 490
      W(1)=ASTER(LOC)
      W(2)=ASTER(LOC+1)
      W(3)=ASTER(LOC+2)
      UW(1)=ASTER(LOC+4)-W(1)
      UW(2)=ASTER(LOC+5)-W(2)
      UW(3)=ASTER(LOC+6)-W(3)
      VW(1)=ASTER(LOC+8)-W(1)
      VW(2)=ASTER(LOC+9)-W(2)
      VW(3)=ASTER(LOC+10)-W(3)
C          WN = (U-W) X (V-W)
      WN(1)=UW(2)*VW(3)-UW(3)*VW(2)
      WN(2)=UW(3)*VW(1)-UW(1)*VW(3)
      WN(3)=UW(1)*VW(2)-UW(2)*VW(1)
C          D = WB . WN
      D=WB(1)*WN(1)+WB(2)*WN(2)+WB(3)*WN(3)
C
C          DETERMINE IF RAY PARALLEL TO PLANE OF TRIANGLE
C
      IF(ABS(D).LE.0.0001)GOTO 490
      WXB(1)=W(1)-XB(1)
      WXB(2)=W(2)-XB(2)
      WXB(3)=W(3)-XB(3)
```

FIG. 95. (Contd.)

```
C         DALPHA = (W-XB) . ( WB X (V-W) )
      DALPHA= WXB(1)*(WB(2)*VW(3)-WB(3)*VW(2))
     1        +WXB(2)*(WB(3)*VW(1)-WB(1)*VW(3))
     2        +WXB(3)*(WB(1)*VW(2)-WB(2)*VW(1))
      ALPHA=DALPHA/D
      IF(ALPHA*(1.-ALPHA).LT.0.0)GOTO 490
C         DBETA = (W-XB) . ( (U-W) X WB )
      DBETA= WXB(1)*(UW(2)*WB(3)-UW(3)*WB(2))
     1        +WXB(2)*(UW(3)*WB(1)-UW(1)*WB(3))
     2        +WXB(3)*(UW(1)*WB(2)-UW(2)*WB(1))
      BETA=DBETA/D
      IF(BETA*(1.-BETA).LT.0.0)GOTO 490
C
      GAMMA=1.-ALPHA-BETA
      IF(GAMMA*(1.-GAMMA).LT.0.0)GOTO 490
C
C         COMPUTE DISTANCE TO INTERSECT WITH TRIANGLE
C
C         DS = (W-XB) . WN
C
      DS=WXB(1)*WN(1)+WXB(2)*WN(2)+WXB(3)*WN(3)
      S=DS/D
      CALL UNIT(WN)
C
C         DIRECT NORMAL INTO ARS FOR ENTRY INTERSECT, OUT OF ARS FOR
C         EXIT INTERSECT
C
      IF(IT-(IT/2)*2.EQ.0)GOTO 410
      WN(1)=-WN(1)
      WN(2)=-WN(2)
      WN(3)=-WN(3)
      D=-D
  410 JSURF=IT
      IF(D.LT.0.0)JSURF=-JSURF
C
C         COMPARE NEW INTERSECT DISTANCE WITH DISTANCES ALREADY IN
C         HIT TABLE
C
C         STORE HITS (LARGEST TO SMALLEST)
C
      IF(NHIT.EQ.0)GOTO 430
      DO 420 I=1,NHIT
      IF(ABS(S-HIT(I)).LE.0.0001)GOTO 470
      IF(S.GT.HIT(I))GOTO 450
  420 CONTINUE
C
  430 NHIT=NHIT+1
      I=NHIT
      IF(NHIT.LE.20)GOTO 440
  435 WRITE(6,901)NBO
      WRITE(6,910)(HIT(I),ISURF(I),I=1,NHIT)
      NHIT=0
      GOTO 700
```

FIG. 95. (Contd.)

```
C
C          IF NEW INTERSECT, STORE HIT
C
  440 HIT(I)=S
      ORMAL(1,I)=WN(1)
      ORMAL(2,I)=WN(2)
      ORMAL(3,I)=WN(3)
      ISURF(I)=JSURF
      GOTO 490
C
C          ADD A HIT TO TABLE WHEN S.GT.HIT(I)
C
  450 J=NHIT
      NHIT=NHIT+1
      IF(NHIT.GT.20)GOTO 435
C
  460 IF(J.LT.I)GOTO 440
      HIT(J+1)=HIT(J)
      ORMAL(1,J+1)=ORMAL(1,J)
      ORMAL(2,J+1)=ORMAL(2,J)
      ORMAL(3,J+1)=ORMAL(3,J)
      ISURF(J+1)=ISURF(J)
      J=J-1
      GOTO 460
C
C          TWO ENTRIES IDENTICAL WHEN S .EQ. HIT(I)
C               IF BOTH RIN OR BOTH ROUT IGNORE
C               IF ONE A RIN AND OTHER A ROUT DELETE ENTRY IN TABLE
C
  470 IF(JSURF*ISURF(I).GT.0)GOTO 490
C
C          DELETE ENTRY
C
      NHIT=NHIT-1
  480 IF(I.GT.NHIT)GOTO 490
      HIT(I)=HIT(I+1)
      ORMAL(1,I)=ORMAL(1,I+1)
      ORMAL(2,I)=ORMAL(2,I+1)
      ORMAL(3,I)=ORMAL(3,I+1)
      ISURF(I)=ISURF(I+1)
      I=I+1
      GOTO 480
C
C          INCREMENT TO TEST NEXT POSSIBLE TRIANGLE
C
  490 LOC=LOC+4
  499 CONTINUE
C
C          ALL POSSIBLE TRIANGLES EXAMINED
C          CHECK FOR AN EVEN NUMBER OF HITS
C
      IF(NHIT.EQ.0)GOTO 700
      IF(NHIT-(NHIT/2)*2.EQ.0)GOTO 500
```

FIG. 95. (Contd.)

```
C
C          ERROR - INCORRECT SEQUENCE OF HITS
C
      WRITE(6,902)NHO,NHIT
      WRITE(6,910)(HIT(I),ISURF(I),I=1,NHIT)
      NHIT=0
      GOTO 700
C
C          CHECK FOR CORRECT SEQUENCE OF EXITS(-) AND ENTRANCES(+)
C
  500 DO 520 I=2,NHIT
      IF(ISURF(I-1)*ISURF(I),GT,0)GOTO 525
  520 CONTINUE
      GOTO 530
C
C          ERROR - INCORRECT SEQUENCE OF HITS
C
  525 WRITE(6,903)NHO,NHIT
      WRITE(6,910)(HIT(I),ISURF(I),I=1,NHIT)
      NHIT=0
      GOTO 700
C
C          LOCATE NEXT ROUT DISTANCE RELATIVE TO CURRENT POSITION OF XP
C
  530 IF(HIT(NHIT-1).GT.0.0)GOTO 540
      NHIT=NHIT-2
      IF(NHIT.LE.0)GOTO 700
      GOTO 530
C
C          CHECK DIRECTION OF NORMAL FOR LARGEST DISTANCE IN HIT TABLE
C          VERIFY THAT NORMAL IS AN EXIT FOR THE ROUT INTERSECT
C
  540 IF(NASC.EQ.-2)GOTO 600
      IF(ISURF(1).LT.0)GOTO 560
      DO 550 I=1,NHIT
      ORMAL(1,I)=-ORMAL(1,I)
      ORMAL(2,I)=-ORMAL(2,I)
      ORMAL(3,I)=-ORMAL(3,I)
      ISURF(I)=-ISURF(I)
  550 CONTINUE
C
C          STORE HIT TABLE IN ASTER ARRAY
C          UNLESS COMPUTING NORMAL DISTANCE
C
  560 LOC=LOCHTS
      DO 570 I=1,NHIT
      ASTER(LOC)=HIT(I)
      ASTER(LOC+1)=ORMAL(1,I)
      ASTER(LOC+2)=ORMAL(2,I)
      ASTER(LOC+3)=ORMAL(3,I)
      LOC=LOC+4
  570 CONTINUE
```

FIG. 95. (Contd.)

```
C
C          CHOOSE CORRECT RIN AND ROUT SET FOR CURRENT POSITION OF XP
C          THIS SECTION IS ALSO USED BY REENTRY ROUTINE
C
  600 IF(NHIT.EQ.0)GOTO 700
      RIN=HIT(NHIT)
      ROUT=HIT(NHIT-1)
      LRT=1
      LRO=1
      NHIT=NHIT-2
      IF(ABS(DIST-ROUT).LE.0.0001)GOTO 600
      IF(DIST.GE.ROUT)GOTO 600
      IF(ABS(RIN-ROUT).LE.0.0001)GOTO 600
      IF(RIN.GT.0.0001)GOTO 800
      RIN=-PINF
      LRT=0
      GOTO 800
C
C          RAY MISSES ARS FROM CURRENT LOCATION
C
  700 RIN=PINF
      ROUT=-PINF
      LRT=0
      LRO=0
      IF(N.EQ.0)NHIT=0
  800 IF(NASC.NE.-2)MASTER(LOCARS+1)=NHIT
      RETURN
      END
```

FIG. 95. (Concluded)

```
      SUBROUTINE TOR
      DIMENSION COEF(4),RT(4),XN(3),XBC(3)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1 LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/DAVIS/IGRID,LOOP,INORM
      EQUIVALENCE(ASTER,MASTER)
C
C1    CHECK FOR PREVIOUS ENTRY
C
      IF(LOOP.NE.KLOOP)GOTO 10
C
C2    THIS IS A REENTRY
C
      NR=MASTER(LOCDA+2)
      IF(NR.LE.2)RETURN
      RIN=ASTER(LOCDA+3)
      ROUT=ASTER(LOCDA+4)
      NR=0
      GOTO 400
C
C3    RETRIEVE LOCATIONS OF TORUS DATA
C
   10 CALL UN2(LOCDA,IV,IN)
      LOC=LOCDA+1
      CALL UN2(LOC,IR1,IR2)
C
C4    COMPUTE INTERMEDIATE VARIABLES NEEDED TO
C     FIND COEFFICIENTS OF QUARTIC EQUATION
C
      XBC(1)=XB(1)-ASTER(IV)
      XBC(2)=XB(2)-ASTER(IV+1)
      XBC(3)=XB(3)-ASTER(IV+2)
      XN(1)=ASTER(IN)
      XN(2)=ASTER(IN+1)
      XN(3)=ASTER(IN+2)
      R1=ASTER(IR1)
      R2=ASTER(IR2)
      IF(NASC.NE.-2)GOTO 20
      RSAVE=0.
      GOTO 30
   20 RSAVE=ABS(DOT(XBC,WB))-R1-R2-R2
      XBC(1)=XBC(1)+RSAVE*WB(1)
      XBC(2)=XBC(2)+RSAVE*WB(2)
      XBC(3)=XBC(3)+RSAVE*WB(3)
   30 WDN=DOT(WB,XN)
      XBCDW=DOT(XBC,WB)
      XBCDN=DOT(XBC,XN)
      XBCXBC=DOT(XBC,XBC)
      R1SQ=R1*R1
      R2SQ=R2*R2
      TERM=XBCXBC-R1SQ-R2SQ
C
C5    COMPUTE COEFFICIENTS
C
      COEF(1)=4.*XBCDW
      COEF(2)=4.*R1SQ*WDN*WDN+4.*XBCDW*XBCDW+2.*TERM
```

FIG. 96. Source Listing, Subroutine TOR

```
      COEF(3)=8.*R1SQ*WDN*XBCDN+4.*XBCDN*TERM
      COEF(4)=4.*R1SQ*XBCDN*XBCDN+TERM*TERM-4.*R1SQ*R2SQ
      CALL QRTIC(COEF,RT,NR)
C
C6    DETERMINE IF 0, 2, OR 4 ROOTS
C
      IF(NR-2)500,100,200
C
C7    TWO ROOTS
C
  100 IF(ABS(RT(1)-RT(2)).GT.0.0001)GOTO 110
      NR=0
      GOTO 500
  110 RT(1)=RT(1)+RSAVE
      RT(2)=RT(2)+RSAVE
      IF(RT(1).LT.RT(2))GOTO 300
      T=RT(1)
      RT(1)=RT(2)
      RT(2)=T
      GOTO 300
C
C8    FOUR ROOTS
C
  200 DO 210 I=1,4
      RT(I)=RT(I)+RSAVE
  210 CONTINUE
C
C9    SORT ROOTS IN ASCENDING ORDER
C
  220 IF(RT(1).LE.RT(2))GOTO 230
      T=RT(1)
      RT(1)=RT(2)
      RT(2)=T
  230 IF(RT(2).LE.RT(3))GOTO 240
      T=RT(2)
      RT(2)=RT(3)
      RT(3)=T
      GOTO 220
  240 IF(RT(3).LE.RT(4))GOTO 250
      T=RT(3)
      RT(3)=RT(4)
      RT(4)=T
      GOTO 230
C
C10   IF RAY TANGENT TO SURFACE ELIMINATE INTERSECTS
C
  250 IF(ABS(RT(2)-RT(3)).GT.0.0001)GOTO 260
      NR=NR-2
      RT(2)=RT(4)
      GOTO 270
  260 IF(ABS(RT(3)-RT(4)).GT.0.0001)GOTO 270
      NR=NR-2
  270 IF(ABS(RT(1)-RT(2)).GT.0.0001)GOTO 280
      NR=NR-2
      RT(1)=RT(3)
      RT(2)=RT(4)
  280 IF(NR.LE.0)GOTO 500
      IF(RT(2).GT.0.0)GOTO 300
      NR=NR-2
```

FIG. 96. (Contd.)

```
      RT(1)=RT(3)
      RT(2)=RT(4)
      GOTO 280
C
  300 IF(NR-2)500,350,310
C
C11   FOUR INTERSECTS, DETERMINE WHICH RIN/ROUT SET REQUIRED
C
  310 IF(ABS(DIST-RT(2)).LE.0.0001)GOTO 320
      IF(DIST.LT.RT(2))GOTO 330
  320 RIN=RT(3)
      ROUT=RT(4)
      NR=0
      GOTO 400
  330 ASTER(LOCDA+3)=RT(3)
      ASTER(LOCDA+4)=RT(4)
  350 RIN=RT(1)
      ROUT=RT(2)
  400 LRI=1
      LRO=1
      IF(RIN.GE.ROUT)GOTO 500
      IF(ABS(RIN-ROUT).LE.0.0001)GOTO 500
      IF(ROUT.LE.0.0)GOTO 500
      IF(RIN.GT.0.0001)GOTO 600
      RIN=-.0001
      LRI=0
      GOTO 600
C
C12   RAY MISSES FROM PRESENT ORIGIN
C
  500 RIN=PINF
      ROUT=-PINF
      LRI=0
      LRO=0
  600 MASTER(LOCDA+2)=NR
      RETURN
      END
C
C
```

FIG. 96. (Concluded)

```
      SUBROUTINE QRTIC(C,R,N)
      DIMENSION C(4),R(4),CC(3),RR(3)
C
C1    SOLVES A POLYNOMIAL EQUATION OF THE TYPE X**4 + C(1)*X**3
C     + C(2)*X**2 + C(3)*X + C(4) = 0 USING THE FERRARI SOLUTION OF
C     THE QUARTIC EQUATION. THE COEFFICIENT OF X**4 IS ASSUMED TO BE 1.
C     R(4) CONTAINS THE ROOTS. N CONTAINS THE NUMBER OF REAL ROOTS.
C     IF THERE ARE 2 REAL ROOTS THEY WILL BE IN R(1) AND R(2), WITH THE
C     COMPLEX ROOTS IN R(3)+- R(4)*I. IF THERE ARE NO REAL ROOTS THE
C     COMPLEX ROOTS ARE IN R(1)+-R(2)*I AND R(3)+-R(4)*I.
C
C     COMPUTE RESOLVENT CUBIC
C
      C1SQ=C(1)*C(1)
      CC(1)=-.5*C(2)
      CC(2)=.25*C(1)*C(3)-C(4)
      CC(3)=.125*( C(4)*(4.*C(2)-C1SQ)-C(3)*C(3) )
      CALL CUBIC(CC,RR,NN)
C
C2    DETERMINE IF POSSIBLE SOLUTION
C
      T=.25*C1SQ-C(2)
      DO 10 I=1,NN
      ROOT=RR(I)
      ASQ=T+ROOT+ROOT
      IF(ABS(ASQ).LE.0.000001)ASQ=0.
      IF(ASQ.LT.0.0)GOTO 10
      BSQ=ROOT*ROOT-C(4)
      IF(ABS(BSQ).LE.0.000001)BSQ=0.
      IF(BSQ.GE.0.0)GOTO 20
   10 CONTINUE
      N=0
      RETURN
C
C3    COMPUTE FIRST TWO ROOTS OF QUARTIC EQUATION
C
   20 TWOAB=C(1)*ROOT-C(3)
      A=SQRT(ASQ)
      B=SIGN(SQRT(BSQ),TWOAB)
      N=0
      REAL=.25*(A+A-C(1))
      DISC=REAL*REAL-ROOT+B
      SQROOT=SQRT(ABS(DISC))
      IF(ABS(DISC).LE.0.000001)DISC=0.
      IF(DISC.LT.0.0)GOTO 30
C
C4    DISCRIMINATE .GE. 0  COMPUTE 2 REAL ROOTS
C
      N=2
      R(1)=REAL+SQROOT
      R(2)=REAL-SQROOT
      GOTO 40
C
C5    DISCRIMINATE .LT. 0  COMPUTE 2 IMAGINARY ROOTS
C
   30 R(3)=REAL
      R(4)=SQROOT
C
C6    COMPUTE LAST TWO ROOTS OF QUARTIC EQUATION
C
```

FIG. 97. Source Listing, Subroutine QRTIC

```
   40 REAL=REAL-A
      DISC=REAL*REAL-ROOT-B
      SQROOT=SQRT(ABS(DISC))
      IF(ABS(DISC).LE.0.000001)DISC=0.
      IF(DISC.LT.0.0)GOTO 50
C
C7    DISCRIMINATE .GE. 0  COMPUTE 2 REAL ROOTS
C
      N=N+2
      R(N)=REAL-SQROOT
      R(N-1)=REAL+SQROOT
      RETURN
C
C8    DISCRIMINATE .LT. 0  COMPUTE 2 IMAGINARY ROOTS
C
   50 R(N+1)=REAL
      R(N+2)=SQROOT
      RETURN
      END
C
C
```

FIG. 97. (Concluded)

```
      SUBROUTINE CUBIC(C,R,N)
      DIMENSION C(3),R(3)
C
C1    COMPUTE ROOTS OF CUBIC EQUATION
C
      C1SQ=C(1)*C(1)
      P=C(2)-C1SQ/3.
      Q=C(3)+C(1)*(2.*C1SQ/27.-C(2)/3.)
      DISC=4.*P*P*P+27.*Q*Q
      C3=C(1)/3.
      IF(ABS(DISC).LE.1.0E-8)DISC=0.
      IF(DISC.LE.0.0)GOTO 10
C
C2    CONDITION FOR 1 REAL AND 2 COMPLEX ROOTS
C
      N=1
      SQROOT=SQRT(DISC/108.)
      HALFQ=.5*Q
      ACU=-HALFQ+SQROOT
      BCU=-HALFQ-SQROOT
      A=SIGN(ABS(ACU)**.333333333333333,ACU)
      B=SIGN(ABS(BCU)**.333333333333333,BCU)
      AB=A+B
      R(1)=AB-C3
      R(2)=-.5*AB-C3
      R(3)=.866025404*(A-B)
      RETURN
C
C3    CONDITION FOR 3 REAL ROOTS
C
   10 N=3
      T=SQRT(ABS(P)/3.)
      TT=T+T
      IF(DISC.EQ.0.0)GOTO 20
      PHI3=ATAN2(SQRT(-DISC/27.),-Q)/3.
      R(1)=TT*COS(PHI3)-C3
      R(2)=TT*COS(PHI3+2.094395103)-C3
      R(3)=TT*COS(PHI3-2.094395103)-C3
      RETURN
C
C4    CONDITION FOR 2 OR 3 EQUAL ROOTS
C
   20 R(1)=SIGN(TT,-Q)-C3
      R(2)=SIGN(T,Q)-C3
      R(3)=R(2)
      RETURN
      END
C
C
```

FIG. 98. Source Listing, Subroutine CUBIC

```
      SUBROUTINE UN2(L,J1,J2)
C
C     UNPACK 2 15-BIT INTEGER DATA ITEMS FROM L WORD IN MASTER ARRAY
C
      COMMON MASTER(10000)
      I3=MASTER(L)
      J1=I3/32768
      J2=I3-J1*32768
      RETURN
      END
C
C
```

FIG. 99. Source Listing, Subroutine UN2

```
      SUBROUTINE UN3(L,J1,J2,J3)
C
C     UNPACK 2 6-BIT AND 1 15-BIT INTEGER DATA ITEMS FROM G1 WORKING
C     STORAGE AT THE L WORD IN THE MASTER ARRAY
C
      COMMON MASTER(10000)
      I3=MASTER(L)
      I2=I3/32768
      J1=I2/64
      J2=I2-J1*64
      J3=I3-I2*32768
      RETURN
      END
C
C
```

FIG. 100. Source Listing, Subroutine UN3

```
      SUBROUTINE OPENK(L,J1,J2,J3)
      COMMON/GTRACK/D1,D2,KHIT,LMAX,TR(200),XBS(3),IRSTRT,IENC,
     1    ITR(200),CA,CE,SA,SE
C
C1    UNPACK 3 12-BIT INTEGER DATA ITEMS FROM COMPONENT LINE-OF-SIGHT
C     STORAGE ARRAY ITR, THE THREE ITEMS ARE
C         / SURFACE NUMBER /  BODY NUMBER   /   NEXT REGION  /
C
      I3=ITR(L)
      I2=I3/4096
      J1=I2/4096
      J2=I2-J1*4096
      J3=I3-I2*4096
      RETURN
      END
C
C
```

FIG. 101. Source Listing, Subroutine OPENK

```
      FUNCTION RAN(M)
      COMMON/RANDM/IRN
C
C1    GIVEN A NUMBER IRN, GENERATE A RANDOM NUMBER BETWEEN 0 AND 1
C
      RAN=URAN31(IRN)
      RETURN
      END
C
C
```

FIG. 102. Source Listing, Subroutine RAN

```
      FUNCTION URAN31(I)
      IF(I)20,10,20
   10 I=11111111
   20 J=I
      J=J*25
      J=J-(J/67108864)*67108864
      J=J*25
      J=J-(J/67108864)*67108864
      J=J*5
      J=J-(J/67108864)*67108864
      A1=J
      I=J
      URAN31=A1/67108864.
      RETURN
      END
C
C
```

FIG. 103. Source Listing, Subroutine URAN31

```
      SUBROUTINE CROSS(ANSWER,FIRST,SECONU)
      DIMENSION ANSWER(3),FIRST(3),SECOND(3)
C
C     COMPUTE CROSS PRODUCT    ANSWER = FIRST X SECOND
C
      ANSWER(1) = FIRST(2)*SECOND(3) - FIRST(3)*SECOND(2)
      ANSWER(2) = FIRST(3)*SECOND(1) - FIRST(1)*SECOND(3)
      ANSWER(3) = FIRST(1)*SECOND(2) - FIRST(2)*SECOND(1)
      RETURN
      END
C
C
```

FIG. 104. Source Listing, Subroutine CROSS

```
      FUNCTION DOT(FIRST,SECOND)
      DIMENSION FIRST(3),SECOND(3)
C
C1    COMPUTE DOT PRODUCT     DOT = FIRST . SECOND
C
      DOT = FIRST(1)*SECOND(1)+FIRST(2)*SECOND(2)+FIRST(3)*SECOND(3)
      RETURN
      END
C
C
```

FIG. 105. Source Listing, Subroutine DOT

```
      SUBROUTINE UNIT(V)
      DIMENSION V(3)
C
C1    COMPUTE UNIT VECTOR (DIRECTION COSINES OF VECTOR)
C
      TEMP = SQRT(DOT(V,V))
      V(1)=V(1)/TEMP
      V(2)=V(2)/TEMP
      V(3)=V(3)/TEMP
      RETURN
      END
C
C
```

FIG. 106. Source Listing, Subroutine UNIT

```
      FUNCTION XDIST(XA,XB)
C
C1    COMPUTE THE DISTANCE BETWEEN TWO GIVEN POINTS XA AND XB
C
      DIMENSION XA(3),XB(3)
      XSUM=0.
      DO 10 I=1,3
      XSUM=XSUM+(XA(I)-XB(I))**2
   10 CONTINUE
      XDIST=SQRT(XSUM)
      RETURN
      END
C
C
```

FIG. 107. Source Listing, Subroutine XDIST

```
      SUBROUTINE DCOSP(XA,XB,WA)
C
C1    COMPUTE THE DIRECTION COSINES FROM POINT XA TO POINT XB
C     AND STORE DIRECTION COSINES IN WA
C
      DIMENSION XA(3),XB(3),WA(3)
      DIS=XDIST(XA,XB)
      DO 10 I=1,3
      WA(I)=(XB(I)-XA(I))/DIS
   10 CONTINUE
      RETURN
      END
C
C
```

FIG. 108. Source Listing, Subroutine DCOSP

```
      SUBROUTINE TROPIC(WP)
C
C1    GENERATE RANDOM DIRECTION COSINES FROM AN ISOTROPIC DISTRIBUTION
C
      DIMENSION WP(3)
   10 X1=RAN (-1)
      X2=RAN (-1)
      X1S=X1**2
      X2S=X2**2
      T=X1S+X2S
      IF(T.GE.1.)GOTO 10
C
C2    COMPUTE THE SINE AND COSINE OF A RANDOM ANGLE PHI
C
      CSPHI=(X1S-X2S)/T
      SNPHI=(2.*X1*X2)/T
      X1=RAN (-1)
      IF(X1.LE..5)SNPHI=-SNPHI
C
C3    COMPUTE THE SINE AND COSINE OF A RANDOM ANGLE THETA
C
      CSTHT=2.*RAN (-1)-1.
      SNTHT=SQRT(1.-CSTHT**2)
C
C4    COMPUTE RANDOM DIRECTION COSINES
C
      WP(1)=SNTHT*SNPHI
      WP(2)=SNTHT*CSPHI
      WP(3)=CSTHT
      RETURN
      END
C
C
```

FIG. 109. Source Listing, Subroutine TROPIC

```
      FUNCTION S(I,N)
      DIMENSION MASTER(10000)
      COMMON ASTER(10000)
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      EQUIVALENCE(MASTER,ASTER)
C
CI    S RETRIEVES COORDINATE OF ANY ONE OF THE 6 SIDES OF AN RPP
C          I IS THE RPP NUMBER      N IS THE SURFACE NUMBER
C
      L=LBASE+12*(I-1)+2*(N-1)
      LL=MASTER(L+1)
      S=ASTER(LL)
      RETURN
      END
C
C
```

FIG. 110. Source Listing, Subroutine S

```
      SUBROUTINE RPP2(LSURF,XP,IRP)
C
C     FIND NUMBER OF ABUTTING RPP TO INTERSECTED SURFACE
C
      DIMENSION XP(3)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
C
      LOC=LBASE+12*(NASC-1)+2*(LSURF+1)
      CALL UN2(LOC,LOCAT,NC)
      IF(NC-1)10,20,30
   10 IRP=0
      RETURN
   20 CALL UN2(LOCAT,IRP,DUM)
      RETURN
   30 M=1
C
      DO 90 I=1,NC
      M=-M
      IF(M.GT.0)GOTO 50
      CALL UN2(LOCAT,I1,I2)
      LOCAT=LOCAT+1
      IRP=I1
      GOTO 70
   50 IRP=I2
   70 LS=(1-LSURF)/2
      DO 80 J=1,3
      IF(J.EQ.LS)GOTO 80
      IF((S(IRP,2*J-1)-XP(J))*(XP(J)-S(IRP,2*J)).LT.0.)GOTO 90
   80 CONTINUE
      RETURN
   90 CONTINUE
      IRP=0
      RETURN
      END
C
C
```

FIG. 111. Source Listing, Subroutine RPP2

```
      SUBROUTINE VOLUM
C
C1    COMPUTE VOLUMES BY REGION IN VOLUME DEFINED BY BOX
C
      DIMENSION VASTER(1000),WAB(3),WTB(3),WOB(3),DSP(3),
     1  XV(3),XT(3),XA(3),XO(3),XP(3),XTEMP(3)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/WALT/LIRFO,NG1ERR
C
  901 FORMAT(3E20.8)
  902 FORMAT(2E20.8)
  903 FORMAT(1H0,10X,6HVERTEX,14X,6HTOP.PT,14X,6HBOT.PT,14X,7HSIDE.PT)
  904 FORMAT(4E20.8)
  905 FORMAT(1H0, 8X,12HDELTA ON TOP,E20.8,10X,10HSIDE DELTA,E20.8)
  906 FORMAT(2I10)
  908 FORMAT(1H0, 2X,18HSTARTING REGION IS,I5)
  909 FORMAT(1H0,16HVASTER OVERWRITE,5X,6HNRMAX=,I5)
  910 FORMAT(I10,E20.8)
  911 FORMAT(1H0,8HBAD CARD/I10,E20.8,14H NOT PROCESSED)
  912 FORMAT(I10,E20.8,5X,E20.8,5X,E9.2)
  913 FORMAT(1H0,5HSUMV=,5X,E20.8)
C
      READ (5,906)IR,NG1ERR
      IF(NG1ERR.LE.0)NG1ERR=25
C
C2    ENTER COORDINATES OF BOX
C
      READ (5,901)(XV(I),I=1,3)
      READ (5,901)(XT(I),I=1,3)
      READ (5,901)(XO(I),I=1,3)
      READ (5,901)(XA(I),I=1,3)
C
C3    ENTER CELL SIZE
C
      READ (5,902)DOD,DT
      WRITE (6,903)
      WRITE (6,904)(XV(J),XT(J),XO(J),XA(J),J=1,3)
      WRITE (6,905)DOD,DT
      WRITE (6,908)IR
      IF(NRMAX.GT.2000)WRITE (6,909)NRMAX
      CALL DCOSP(XV,XT,WTB)
      CALL DCOSP(XV,XO,WOB)
      CALL DCOSP(XV,XA,WAB)
      XVDIS=XDIST(XV,XA)
      TESTON=0.
      TESTOV=0.
      XTEMP(1)=0.
      DO 10 I=1,NRMAX
      VASTER(I)=0.
   10 CONTINUE
      JIR=IR
      IRJ=IR
C
C4    COMPUTE NUMBER OF HORIZONTAL AND VERTICAL CELLS
C
```

FIG. 112. Source Listing, Subroutine VOLUM

```
      N2=XDIST(XV,XOI/DOD+1.
      N1=XDIST(XV,XT)/DT+1.
C
C5    TRACE RAYS FROM LOWER RIGHT CORNER OF EACH CELL
C
      DO 300 J=1,N2
      DO 100 I=1,3
      DSP(I)=WTB(I)*DT
      XB(I)=XV(I)
      WB(I)=WAB(I)
  100 CONTINUE
      S1=0.
      IR=JIR
C
C6    TRACE ALL RAYS FROM COLUMN OF CELLS
C
      DO 200 I=1,N1
      NASC=-1
C
C7    TRACE RAY THROUGH BOX VIA SUBROUTINE G1
C
  110 CALL G1(S1,IRPRIM,XP)
      IF(IERR.GE.NG1ERR)GOTO 400
      VASTER(IR)=VASTER(IR)+S1
      IF(DIST.GE.XVDIS)GOTO 115
      IF(IRPRIM.LE.0)GOTO 120
      IR=IRPRIM
      GOTO 110
  115 VASTER(IR)=VASTER(IR)-(DIST-XVDIS)
  120 XTEMP(1)=WB(1)
      XTEMP(2)=WB(2)
      XTEMP(3)=WB(3)
      IR=JIR
      TESTDN=TESTDN-DT
      IF(TESTDN.GT.0.)GOTO 180
      WB(1)=WTB(1)
      WB(2)=WTB(2)
      WB(3)=WTB(3)
      NASC=-1
C
C8    DETERMINE REGION OF NEXT ORIGIN OF RAY IN COLUMN
C
      CALL G1(S1,IRPRIM,XP)
      IF(IERR.GE.NG1ERR)GOTO 400
      IF(S1-DT)130,160,170
  130 IR=IRPRIM
      JIR=IR
      CALL G1(S1,IRPRIM,XP)
      IF(IERR.GE.NG1ERR)GOTO 400
      IF(DIST-DT)140,160,170
  140 IF(IRPRIM)150,210,130
  150 STOP
  160 IR=IRPRIM
      JIR=IR
  170 TESTDN=S1
C
C10   SHIFT ORIGIN OF RAY TO NEXT CELL IN COLUMN
C
  180 DO 190 JI=1,3
      WB(JI)=XTEMP(JI)
```

FIG. 112. (Contd.)

```
      XB(JI)=XB(JI)+DSP(JI)
  190 CONTINUE
  200 CONTINUE
C
C11   ONE COLUMN OF CELLS COMPLETE - SHIFT TO NEXT COLUMN
C
  210 NASC=-1
      DO 220 I=1,3
      WB(I)=WOB(I)
      XB(I)=XV(I)
  220 CONTINUE
      JIR=IRJ
      IR=JIR
      TESTON=0.
      TESTOV=TESTOV-DOD
      IF(TESTOV)230,230,280
C
C12   DETERMINE REGION OF FIRST ORIGIN OF NEXT COLUMN
C
  230 CALL G1(S1,IRPRIM,XP)
      IF(IERR.GE.NG1ERR)GOTO 400
      IF(S1-DOD)240,260,270
  240 IR=IRPRIM
      IRJ=IR
      CALL G1(S1,IRPRIM,XP)
      IF(IERR.GE.NG1ERR)GOTO 400
      IF(DIST-DOD)250,260,270
  250 IF(IRPRIM)255,400,230
  255 STOP
  260 IR=IRPRIM
      IRJ=IR
  270 TESTOV=S1
C
C13   SHIFT ORIGIN OF NEXT RAY TO FIRST ORIGIN OF NEXT COLUMN OF CELLS
C
  280 DO 290 I=1,3
      XA(I)=XA(I)+WOB(I)*DOD
      XV(I)=XV(I)+WOB(I)*DOD
      XT(I)=XT(I)+WOB(I)*DOD
  290 CONTINUE
      JIR=IR
  300 CONTINUE
C
C14   ALL RAY DISTANCES THROUGH EACH REGION IN BOX ACCUMULATED
C
  400 READ (5,910)IR1,VR
      IF(IERR.GE.NG1ERR)GOTO 500
      IF(IR1.LE.0)IR1=NRMAX+1
      SUMV=0.
C
C15   COMPUTE VOLUME OF EACH REGION IN BOX
C
      DO 450 I=1,NRMAX
      VASTER(I)=VASTER(I)*DOD*DT
      IF(I-IR1)410,430,420
  410 WRITE (6,910)I,VASTER(I)
      GOTO 440
  420 WRITE (6,911)IR1,VR
      READ (5,910)IR1,VR
      GOTO 410
```

FIG. 112. (Contd.)

```
C
C16    COMPUTE PERCENT ERROR FOR PRE-COMPUTED VOLUME OF GIVEN REGION
C
  430 XPERC=100.*(VASTER(I)/VR-1.)
      WRITE (6,912)I,VASTER(I),VR,XPERC
      VASTER(I)=VR
      READ (5,910)IR1,VR
C
C17    COMPUTE TOTAL VOLUME OF ALL REGIONS WITHIN BOX
C
  440 SUMV=SUMV+VASTER(I)
  450 CONTINUE
      WRITE (6,913)SUMV
  500 IERR=0
      RETURN
      END
C
C
```

FIG. 112. (Concluded)

```
      SUBROUTINE AREA
      DIMENSION XP(3),WP(3),XBS(3),CONVRT(4,4),TYPEUN(4)
      COMMON ASTER(10000)
      COMMON/PAREM/XB(3),WB(3),IR
       COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1  LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/CAL/NIR,SLOS,ANGLE,NTYPE,SSPACE,L,XS(3),WS(3),
     1  TRAVEL,SN,V,H,IVIH
      COMMON/WALT/LIRFO,NG1ERR
      COMMON/CELL/CELSIZ
      COMMON/ENGEOM/LEGEOM
C
  901 FORMAT(7I10,6X,2A2)
  902 FORMAT(6E12.5)
  908 FORMAT(1H0,22HMEMORY OVERLAP IN AREA,5X,7HLEGEOM=,I6,
     1      5X,6HLAREA=,I6,5X,6HLIRFO=,I6)
  909 FORMAT(1H0,13HERROR IN AREA,5X,9HICODE = 0)
  910 FORMAT(1H0,8HAZIMUTH=,F10.3,5X,10HELEVATION=,F10.3)
  911 FORMAT(1H0,12HCELL SIZE IS,F4.1,1X,1HX,F4.1,1X,A2,1H.,10X,
     1      12HAREAS IN SQ.,1X,A2,1H.)
  912 FORMAT(1H0,5HICODE,19X,4HAREA/)
  913 FORMAT(I5,15X,F12.5)
  914 FORMAT(1H0,15HPRESENTED AREA=,F12.5)
  915 FORMAT(1H0,18HNUMBER OF CELLS IS,I5,10X,
     1      22HNUMBER OF CELLS HIT IS,I5)
C
      DATA HHIN,HHFT,HHCM,HHMB,HHBB/2HIN,2HFT,2HCM,2HM ,2H  /
      TYPEUN(1)=HHIN
      TYPEUN(2)=HHFT
      TYPEUN(3)=HHCM
      TYPEUN(4)=HHMB
      CONVRT(1,1)=1.
      CONVRT(1,2)=.00694444444444
      CONVRT(1,3)=6.451625806
      CONVRT(1,4)=.0006451625806
      CONVRT(2,1)=144.
      CONVRT(2,2)=1.
      CONVRT(2,3)=929.0341161
      CONVRT(2,4)=.09290341161
      CONVRT(3,1)=.15499969
      CONVRT(3,2)=.001076386736
      CONVRT(3,3)=1.
      CONVRT(3,4)=.0001
      CONVRT(4,1)=1549.9969
      CONVRT(4,2)=10.7636736
      CONVRT(4,3)=10000.
      CONVRT(4,4)=1.
      BLANK=HHBB
C
C1    COMPUTE AND INITIALIZE AREA FOR STORING PRESENTED AREA
C     BY COMPONENT CODE
C
      LAREA=LIRFO-1000
      IF(LAREA.GE.LEGEOM)GOTO 10
      WRITE (6,908)LEGEOM,LAREA,LIRFO
      STOP
   10 LAREA1=LIRFO-1
      DO 20 L=LAREA,LAREA1
      ASTER(L)=0.
```

FIG. 113. Source Listing, Subroutine AREA

```
   20 CONTINUE
C
C2     READ GRID INPUT PARAMETERS
C
       READ (5,901)NX,NY,IRSTRT ,IENC,NG1ERR,NSTART,NEND,CELLUN,AREAUN
       READ (5,902)A,E,ENGTH,ZSHIFT,GROUND
       READ (5,902)XSHIFT,YSHIFT,CELSIZ
C
C3     INITIALIZE PARAMETERS NOT SET BY INPUT
C
       IF(IRSTRT .LE.0)IRSTRT=1
       IF(CELSIZ .LE.0.)CELSIZ=4.
       IF(NSTART.LE.0)NSTART=1
       IF(NG1ERR.LE.0)NG1ERR=25
       IF(AREAUN.EQ.BLANK)AREAUN=HHIN
       IF(CELLUN.EQ.BLANK)CELLUN=HHIN
C
C4     DETERMINE MEASUREMENT UNITS AND COMPUTE GRID CELL AREA
C
       DO 30 I=1,4
       IF(CELLUN.EQ.TYPEUN(I))GOTO 40
   30 CONTINUE
   40 DO 50 J=1,4
       IF(AREAUN.EQ.TYPEUN(J))GOTO 60
   50 CONTINUE
   60 AREAC=CELSIZ*CELSIZ*CONVRT(I,J)
C
       RADIAN=.017453292519943
       AR=A*RADIAN
       ER=E*RADIAN
       SA=SIN(AR)
       CA=COS(AR)
       SE=SIN(ER)
       CE=COS(ER)
       KL=NX*NY
       NHIT=0
C
C5     PROCESS KL CELLS IN GRID PLANE
C
       DO 200 KK=NSTART,KL
       WB(1)=-CE*CA
       WB(2)=-CE*SA
       WB(3)=-SE
C
C6     COMPUTE ROW AND COLUMN NUMBER OF GRID CELL
C
       II=((KK-1)/NX)+1
       J=KK-(II-1)*NX
C
       CELL2=.5*CELSIZ
       V=FLOAT((NY/2)-II)*CELSIZ +CELL2
       VREF=V+CELL2
       H=FLOAT((NX/2)- J)*CELSIZ +CELL2
       HREF=H+CELL2
       IV=RAN(-1)*10.
       IH=RAN(-1)*10.
       IVIH=10*IH+IV
C
C7     COMPUTE RANDOM POINT WITHIN GRID CELL
C
```

FIG. 113. (Contd.)

```
      V=V+CELSIZ *FLOAT(IV)/10.+CELSIZ /20.
      H=H+CELSIZ *FLOAT(IH)/10.+CELSIZ /20.
C
      XBS(1)=XSHIFT-V*CA*SE-H*SA
      XBS(2)=YSHIFT-V*SA*SE+H*CA
      XBS(3)=ZSHIFT+V*CE
      CALL TROPIC(WP)
      XBS(1)=XBS(1)+WP(1)*1.0E-4
      XBS(2)=XBS(2)+WP(2)*1.0E-4
      XBS(3)=XBS(3)+WP(3)*1.0E-4
C
C8    CONVERT GRID PLANE COORDINATES TO COORDINATES OF TARGET
C
      XB(1)=XBS(1)-ENGTH*WB(1)
      XB(2)=XBS(2)-ENGTH*WB(2)
      XB(3)=XBS(3)-ENGTH*WB(3)
      IF(XB(3).LE.GROUND)GOTO 200
C
C9    TRACE RAY TO FIRST TARGET COMPONENT HIT
C
      IR=IRSTRT
      NASC=-1
  110 CALL G1(S1,IRPRIM,XP)
      IF(IERR.GE.NG1ERR)RETURN
      IF(IRPRIM.LT.0)GOTO 200
      IF(NASC.LE.NRPP)IRPRIM=0
      IF(IRPRIM.EQ.0)GOTO 200
      LOC=LIRFO+IRPRIM-1
      CALL UN2(LOC,ICODE,IDENT)
      IDENT=IDENT-1
      IF(IDENT-(IDENT/10)*10.EQ.0)GOTO 120
      IR=IRPRIM
      GOTO 110
  120 IF(ICODE.NE.0)GOTO 130
      WRITE (6,909)
      GOTO 200
  130 LOC=LAREA+ICODE-1
      ASTER(LOC)=ASTER(LOC)+AREAC
      NHIT=NHIT+1
  200 CONTINUE
C
C10   PRINT RESULTS
C
      WRITE (6,910)A,E
      WRITE (6,911)CELSIZ, CELSIZ, CELLUN,AREAUN
      WRITE (6,912)
      SUMA=0.
      DO 250 I=1,999
      LOC=LAREA+I-1
      IF(ASTER(LOC).EQ.0.)GOTO 250
      WRITE (6,913)I,ASTER(LOC)
      SUMA=SUMA+ASTER(LOC)
  250 CONTINUE
      WRITE (6,914)SUMA
      WRITE (6,915)KL,NHIT
      RETURN
      END
C
C
```

FIG. 113. (Concluded)

```
      SUBROUTINE TESTG
C
C1    TRACE A RAY BETWEEN TWO GIVEN POINTS XB TO XBF
C
      DIMENSION XP(3),XBF(3)
      COMMON/PAREM/XB(3),WB(3),IR
      COMMON/GEOM/LBASE,RIN,ROUT,LRI,LRO,PINF,IERR,DIST
      COMMON/UNCGEM/NRPP,NTRIP,NSCAL,NBODY,NRMAX,LTRIP,LSCAL,LREGD,
     1 LDATA,LRIN,LROT,LIO,LOCDA,I15,I30,LBODY,NASC,KLOOP
      COMMON/WALT/LIRFO,NG1ERR
C
  901 FORMAT(2I10)
  902 FORMAT(1H0,22HNUMBER OF SPECIAL RAYS,I5)
  903 FORMAY(3E15.7,3I15)
  904 FORMAT(1H0,5HSTART,5X,4H XB=,3E15.7,8H IRSTRT=,I5/
     1     4H END,7X,4HXBF=,3E15.7,8H  IRFIN=,I5)
  905 FORMAT(1H0,3HWB=,3E15.7,5X,6HRANGE=,E15.7)
  906 FORMAT(1H0,8X,2HIR,4X,6HIRPRIM,12X,2HS1,13X,2HXP,13X,2HYP,
     1     13X,2HZP,12X,4HDIST)
  907 FORMAT(2I10,5X,5E15.7)
  908 FORMAT(1H0,21HTROUBLE IN REGION IR=,I10)
C
C2    ENTER NUMBER OF RAYS
C
      READ (5,901)NRAYS,NG1ERR
      WRITE (6,902)NRAYS
      IF(NG1ERR.LE.0)NG1ERR=25
C
C3    TRACE GIVEN NUMBER OF RAYS
C
      DO 50 IRAY=1,NRAYS
C
C4    ENTER POINT COORDINATES AND REGION OF EACH
C
      READ (5,903)XB,IRSTRT
      READ (5,903)XBF,IRFIN
      WRITE (6,904)XB,IRSTRT,XBF,IRFIN
      RANGE=XDIST(XB,XBF)
      CALL DCOSP(XB,XBF,WB)
      WRITE (6,905)WB,RANGE
      IR=IRSTRT
      NASC=-1
      WRITE (6,906)
C
C5    TRACE RAY TO NEXT REGION INTERSECT
C
   10 CALL G1(S1,IRPRIM,XP)
      IF(IERR.GE.NG1ERR)GOTO 60
      WRITE (6,907)IR,IRPRIM,S1,XP,DIST
      IF(DIST.GE.RANGE)GOTO 30
      IF(IRPRIM.LE.0)GOTO 20
      IR=IRPRIM
      GOTO 10
C
   20 WRITE (6,908)IR
      GOTO 50
   30 IF(IR.NE.IRFIN)GOTO 20
   50 CONTINUE
   60 IERR=0
      RETURN
      END
C
C
```

FIG. 114. Source Listing, Subroutine TESTG

```
SAMPLE INPUT
            1                                  24         12
-10000.        10000.      -10000.       10000.      -10000.      10000.
2  BOX      75.        -36.        12.       -150.        0.          0.       BODY
2           0.          72.        0.          0.         0.          36.
3  BOX      74.        -35.        13.       -148.        0.          0.       (1.0)
3           0.          70.        0.          0.         0.          34.
4  ARB      75.        -36.        12.         75.        36.         12.      FRONT
4           75.         36.        48.         75.       -36.         48.
4           100.        0.         12.        100.        0.          12.
4           100.        0.         12.        100.        0.          12.
4           1234 6435 6128 6237 7415 7415
5  ARB     -75.        -36.        12.        -75.        36.         12.      REAR
5          -75.         36.        48.        -75.       -36.         48.
5          -100.       -24.        12.       -100.        24.         12.
5          -100.        24.        20.       -100.       -24.         20.
5           1234 5678 3487 1265 2376 1485
6  ELL      20.         0.         48.        -20.        0.          48.      BUBBLE
6           50.
7  ELL   7  0.          0.         48.         24.        0.          0.       (1.0)
7           14.
8  RCC      60.        -36.        12.        -0.         8.          0.       WHEEL
8           12.
9  RCC      60.         36.        12.         0.        -8.          0.       WHEEL
9           12.
10 RCC     -60.        -36.        12.         0.         8.          0.       WHEEL
10          12.
11 RCC     -60.         36.        12.         0.        -8.          0.       WHEEL
11          12.
12 BOX     -70.        -20.        15.         40.        0.          0.       ENGINE
12          0.          40.        0.          0.         0.          30.
13 RAW     -70.        -20.        45.         0.         0.         -10.      (ENGINE)
13          0.          10.        0.          40.        0.          0.
14 RAW     -70.         20.        45.         0.         0.         -10.      (ENGINE)
14          0.         -10.        0.          40.        0.          0.
15 ARB     -70.        -10.        45.        -70.        10.         45.      (ENGINE)
15         -70.         0.         35.        -70.        0.          35.
15         -30.        -10.        45.        -30.        10.         45.
15         -30.         0.         35.        -30.        0.          35.
15          3124 7658 1375 2376 1265 1265
16 ARS
16                      4          5
16         -70.        -20.        15.        -70.       -20.         15.              1
16         -70.        -20.        15.        -70.       -20.         15.              2
16         -70.        -20.        15.                                                 3
16         -70.        -20.        15.        -70.       -10.         15.              4
16         -70.        -10.        25.        -70.       -20.         35.              5
16         -70.        -20.        15.                                                 6
16         -30.        -20.        15.        -30.       -10.         15.              7
16         -30.        -10.        25.        -30.       -20.         35.              8
16         -30.        -20.        15.                                                 9
16         -30.        -20.        15.        -30.       -20.         15.              10
16         -30.        -20.        15.        -30.       -20.         15.              11
16         -30.        -20.        15.                                                 12
17 ARS
17                      5          4
17         -70.         20.        15.        -70.        20.         15.              1
17         -30.         20.        15.        -30.        20.         15.              2
```

FIG. 115. Listing, Sample Problem Data Deck

```
17          -70.      20.      15.      -70.      10.      15.               3
17          -30.      10.      15.      -30.      20.      15.               4
17          -70.      20.      15.      -70.      10.      25.               5
17          -30.      10.      25.      -30.      20.      15.               6
17          -70.      20.      15.      -70.      20.      35.               7
17          -30.      20.      35.      -30.      20.      15.               8
17          -70.      20.      15.      -70.      20.      15.               9
17          -30.      20.      15.      -30.      20.      15.              10
18 REC       0.        0.      24.        0.       0.      28.     TRUNK
18           0.       7.5       0.        5.       0.       0.
19 SPH       0.        0.      52.        5.                       HEAD
20 TEC       0.      -7.5      49.       20.       0.     -12.     ARM
20           0.        0.       3.        0.       2.       0.
20           2.
21 TEC       0.       7.5      49.       20.       0.     -12.     ARM
21           0.        0.       3.        0.       2.       0.
21           2.
22 TRC      -2.      -4.5      27.       32.       0.     -12.     LEG
22           3.        2.
23 TRC      -2.       4.5      27.       32.       0.     -12.     LEG
23           3.        2.
24 TOR      21.5       0.      37.        1.       0.       0.     STEERING
24           8.        1.                                          WHEEL
25 ARB      21.5      -6.      33.5      21.5      6.      33.5    CENTER
25          21.5       0.      44.       40.       0.      37.     STEERING
25          21.5      -6.      33.5      21.5      6.      33.5    WHEEL
25          21.5       0.      44.       40.       0.      37.
25          3127 2146 4328 1345 3127 3127
    1        1      -2      -4      -5      -6      -8      -9     -10     -11
    2 OR     2      -3      -7      -8      -9     -10    -110R    40R      5
    3        6      -7      -2
    4        8
    5        9
    6       10
    7       11
    8 OR     3     -18     -19     -20     -21     -22     -23     -24     -25
            -8      -9     -10     -11    -120R     7     -18     -19     -20
           -21     -24    -250R    130R    140R    150R    160R     17
    9        3      12     -13     -14     -15     -16     -17
   10 OR   180R    190R    200R    210R    220R     23
   11        3      24     -25
   12        3      25
   -1
         1                      01                OUTSIDE AIR
         2          100                           BODY
         3          101                           BUBBLE
         4          151                           WHEEL RIGHT FRONT
         5          152                           WHEEL LEFT FRONT
         6          153                           WHEEL RIGHT REAR
         7          154                           WHEEL LEFT REAR
         8                      02                AIR INSIDE
         9          200                           ENGINE
        10          300                           MAN
        11          400                           STEERING WHEEL
        12          401                           STEERING SHAFT

    2
        37          71          1         1
0.            0.          200.        0.          -500.
0.             0.         2.
        51          37          1         1                                 1
90.           0.          200.        0.          -500.
```

FIG. 115 (Concluded)

☆U.S. GOVERNMENT PRINTING OFFICE: 1971—769-003/18